공학석사 학위논문

캐주얼 게임 유저 이탈 예측

2016년 8월

서울대학교 대학원

융합과학부 디지털정보융합전공

김 승 욱

캐주얼 게임 유저 이탈 예측

지도 교수 이 원 종

이 논문을 공학석사 학위논문으로 제출함

2016년 8월

서울대학교 대학원

융합과학부 디지털정보융합전공

김 승 욱

김승욱의 공학석사 학위논문을 인준함

2016년 8월

위 원 장 강 남 준 (인)

부위원장 이 교 구 (인)

위 원 이 원 종 (인)

초 록

최근 게임 시장의 규모가 커짐에 따라, 게임 시장 중 가장 큰 비중을 차지하고 있는 캐주얼 게임은 점차 각광 받는 산업 분야로 자리 잡고 있다. 이에 많은 게임이 시장에 출시되고 있으며, 모든 게임 업계에서는 수익을 극대화하기 위한 유저 획득에 대해서 신경을 쓰고 있다. 그러나 최근 들어 게임 산업계에서는 추가 유저의 획득에 한계가 있다는 것을 깨닮고, 신규 유저의 유입보다 기존 유저 유지(Retention)의 중요성을 점차 부각시키고 있다.

본 연구에서는 이러한 상황을 연구자가 직접 개발한 모바일 캐주얼 게임을 포함하여, 총 3개의 캐주얼 게임의 데이터를 통해서 살펴본다. 유저의 유입과 이탈의 흔적이 그대로 남아 있는 데이터를 통해서 유저 이탈에 대해서 정량적 연구를 수행하여 게임 산업에서 관심을 두고 있는 유저 유지 분야에 기여하고자 한다. 본 연구에서 하고자 하는 유저 이탈 예측 연구는 매출 상승에 직결될 뿐만 아니라 객관적인 데이터 기반의 insight를 제공하는 데에 중요성이 있다.

본 연구에서는 총 3종의 캐주얼 게임 데이터를 통해서 유저 이탈 예측 모델을 제안하고, 기존 유저 이탈 예측 연구에 대한 한계점을 개선하고자하며, 이탈 예측 연구의 일련의 과정을 통해 insight를 찾는데 연구의 목적이 있다.

이를 위해 본 연구에서는 유저들의 데이터를 가지고 유저 이탈 예측 모델을 설계하고, 예측 모델에 사용되는 Feature들의 특성을 분석하고 이탈 예측 모델 성능에 끼치는 영향을 살펴보도록 한다. 또한, 이탈 예측에 있어서 관찰 기간 및 이탈예측 기간이 예측 성능에 끼치는 영향을 살펴보도록 하고, 알고리즘 별 예측 성능을 비교 및 분석하기로 한다.

본 연구에서는 유저 이탈에 대해 정의 하고 데이터 전처리(data preprocess) 단계를 거쳐 연구에 사용할 원본 데이터를 생성한 후, 데이터 분석을 위한 알고리즘에 적용하기 적합하도록 가공한다. 이후 게임별 공통 Feature 10개와 특정 게임만 적용 가능한 전용 Feature 4개를 활용하여, Gradient boosting, Logistic regression, Random forest 알고리즘 별 예측 모델을 설계한다. 이후 10-fold cross

validation을 통해 ROC(Receiver operating characteristic) curve의 AUC(Area under the curve) 로 유저 이탈 예측 모델의 성능을 평가한다.

실험 결과 activeDuration, playCount feature가 세개의 게임 모두 예측 모델 성능에 큰 영향을 끼치고, 그 밖에 bestScore, consecutivePlayRatio, worstScore, bestScoreIndex, purchaseCount, bestPurchase feature들이 예측 성능에 추가적인 영향을 주었다. 또한, 관찰 기간이 길어지고, 이탈예측 기간이 짧아질수록 예측 성능이 좋아지며, 관찰 기간과 이탈예측 기간이 전체 기간 중 초반 기간 이전일 경우 두 기간에 따른 예측 성능의 변화가 크기 때문에, 초반 기간 이후로 관찰 기간과 이탈예측 기간을 정의해야 안정적인 예측 결과가 나온다. 그리고 세개의 알고리즘 중 Gradient boosting 알고리즘이 가장 좋은 예측성능을 보였지만, 나머지 두개 알고리즘과의 성능차이는 미미했다.

주요어: 캐주얼 게임, 유저 이탈 예측, 게임 유저 분석, 그라디언트 부스팅, 로지스틱 회귀, 랜덤 포레스트
학 번: 2012-23860

목 차

공식 목차

표 목차

그림 목차

제 1 장 서 론

제 1 절 연구의 배경

미래창조과학부의 조사에 따르면 2015년 국내 가구 스마트폰 보유율은 2013년부터 6.7%씩 지속적으로 상승되어 86.4%에 달한다고 한다. 이에 비해 최근 3년 동안의 가구 PC보유율은 80.6%에서 77.1%로 떨어져 현재 가구 스마트폰 보유율이 PC보유율을 앞선 상태이다. 이는 이메일, 인터넷쇼핑, 온라인뱅킹 등 기존에 PC로 하던 일들을 스마트폰의 활용이 대체하면서 PC의 수요가 줄어들고 있기 때문으로 풀이된다[1,2,3].

이런 변화는 게임 분야에서도 마찬가지로 이루어지고 있다. 2011년도 한국콘텐츠진흥원의 게임이용자조사 연구보고서에 따르면 주로 이용하는 게임분야는 전체 응답자의 67.1%를 차지한 온라인게임이 1순위를 차지하였으며, 모바일게임은 15.3%, PC게임은 8.6%를 차지하였다[4]. 그러나 최근 스마트폰의 보급이 증가됨에 따라 모바일 게임 플랫폼이 기존의 게임 플랫폼들을 대체하기 시작했다[1, 5]. 2015년도 한국콘텐츠진흥원 연구보고서의 설문에 따르면 최근 이용하고 있는 게임 분야(복수응답)로 '모바일게임'이 86.2%로 가장 높고, 다음으로 '온라인게임'(60.3%), 'PC용 패키지게임'(20.9%), '휴대용 콘솔게임'(9.9%), '비디오 콘솔게임'(9.5%), '아케이드게임'(8.1%) 순으로 나타났다[5] [그림 1].

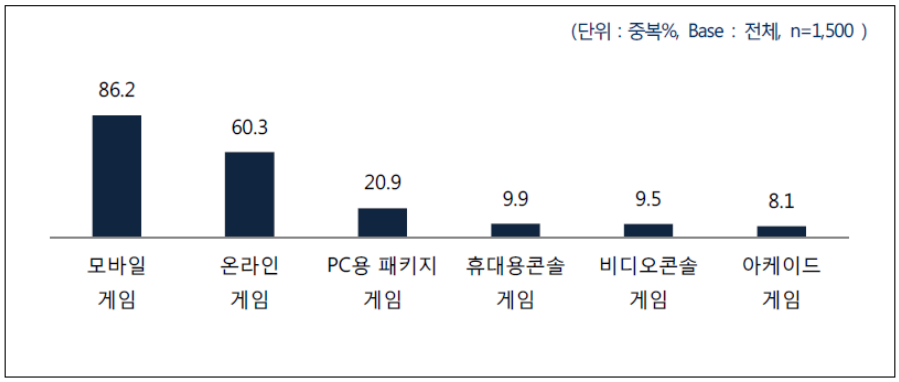

그림 1 게임 이용 분야 (복수응답)

또한, 2014년도 한국콘텐츠진흥원 연구보고서에 따르면 모바일게임 내에서 주로 이용하는 게임에 대한 선호도에 대해 (복수응답) ‘캐주얼게임’이 54.2%로 가장 높게 나타났고 ‘경영/건설/육성 시뮬레이션’(24.8%), ‘웹/보드게임’(24.2%), ‘러닝게임’(19.8%) 등의 순으로 나타났다[6][그림 2].

그림 2 모바일게임 주이용 장르(1+2순위 응답 기준)

위 보고서의 통계에와 같이 전체 게임 분야에서 모바일 게임이 가장 강세로 나타났으며 그 중 캐주얼 게임의 선호도가 가장 높게 나타났다. 국내외 대표적인 캐주얼 게임으로는 ‘Angrybird(2010)’, ‘Anipang(2012)’, ‘Candycrushsaga(2012)’등이 있고 이런 게임들이 선두가 되어 캐주얼 게임 시장을 이끌어 왔다. 위와 같은 캐주얼 게임은 단순하고 쉬운 조작감으로 언제 어디서든 편리하게 할 수 있다는 장점으로 많은 유저들을 끌어모았다.

이러한 캐주얼 게임은 점점 산업에서 주목받고 있으며, 캐주얼 게임을 포함한 여러 게임들이 시장에 지속적으로 나오고 있다. 이에

따라 게임에서의 수익 상승도 주요 관심사로 떠오르고 있는데, 게임에서의 수익 상승을 위해 많은 수의 게임 업체들이 다양한 광고 플랫폼을 이용하여 게임의 설치 및 유저의 유입을 늘리기위해 노력하고 있다.

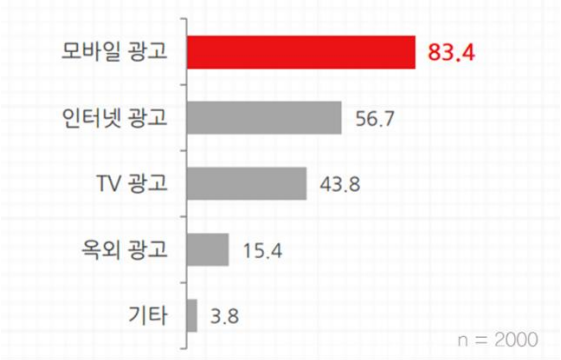

그림 3 모바일게임 설치에 영향을 준 광고 매체

2014년 나스미디어의 NPR(Netizen Profile Research)에 따르면 모바일 게임 설치에 영향을 준 광고 매체의 83.4%가 모바일광고, 56.7%가 인터넷 광고로 나타났고, 나머진 'TV광고'(43.8%), '옥외 광고'(15.4%) 순으로 나타났다[7][그림 3]. 이와 같이 게임 유저들을 획득하기위해 모바일 및 인터넷 광고 플랫폼이 가장 효과적인 채널로 여겨지고 있고, 많이 사용되고 있다.

그림 4 모바일 및 인터넷 광고의 비즈니스 모델

[그림 4]에서와 같이 모바일 및 인터넷 광고 플랫폼에서는 플랫폼의 특성에 맞게 다양한 광고에 맞는 비즈니스 모델을 제공해 주고 있다. 예를 들어 CPI(Click-per-install), CPA(Click-per-active), CPL(Click-per-like), CPF(Click-per-follow), CPV(Click-per-view)등이 제공되고 있는데, 이러한 광고 플랫폼은 유저의 초기 유입에

목표를 맞추어 서비스 되고 있다. 그러나 게임 유저의 수가 충분히 모인 시점에서는 유저의 유입도 중요하지만 유저의 유지(Retention)도 신경을 써야 한다. 특히 구매량이 많은 상위권 유저들의 이탈은 전체 게임 매출에 큰 영향을 준다[8]. 최근 들어 이러한 유저의 유지에 주의를 기울이는 업체들도 생겨나고 있는데, 이런 업체들은 게임 내에 분석 라이브러리를 내장하여 유저의 게임 정보를 수집하고 분석해줌으로써 유저의 유지 관리를 해주고 있다[그림 5]. 유저의 수를 효과적으로 유지하기 위해서 위와 같이 업체들이 많은 노력을 하고 있는데 반해 학계에서의 관련 분야 연구는 상대적으로 부족한 상황으로 보인다.

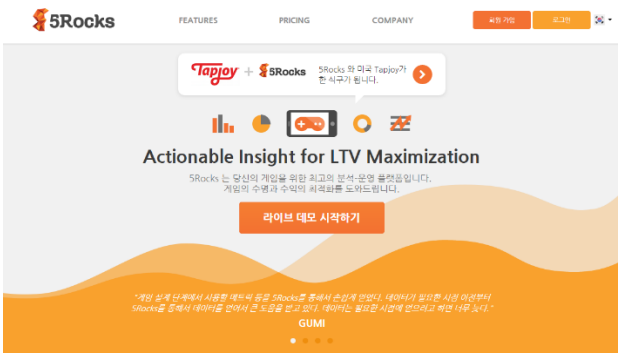

그림 5 게임 분석 업체 화면

본 연구에서 유저 유지 연구를 위해 3가지의 캐주얼 게임을 분석하고, 유저 유지에 관한 연구를 진행하였다. 연구에서 사용한 데이터는 총 3종의 게임에서 추출한 데이터이다.

그림 6 게임화면: Crap dodging (똥피하기)

첫번째 게임인 'Crap dodging(똥피하기)' 는 연구자 본인이 직접 개발한 게임으로써 2011년 6월 Google play에 출시된 Android 전용 모바일 캐주얼 액션 게임이다[그림 6]. 이는 하늘에서 내려오는 장애물을 피해서 높은 점수를 기록 하는 것이 목표인 게임이다.

이 게임은 출시 당시 많은 인기가 있어서 Google play에 캐주얼 게임 분야 상위 2위 랭킹을 기록하였다. 그러나 유저의 유입이 많은 만큼 유저의 이탈도 많아져 [그림 7] 과 같이 시간이 지날수록 일별 플레이 수가 급격히 감소하였다.

그림 7 똥피하기1 일별 유저 플레이 수

두번째 게임은 'Delivery outlaw' 이다. 2014년 5월 iOS Appstore와 Google Play에 출시 되었으며 모바일 캐주얼 레이싱 게임이다[그림 8]. 주어진 연료를 가지고 목적지까지 도착하고, 획득한 게임머니로 탈것을 업그레이드한다. 출시 초반엔 iOS Appstor와 Google Play에 상위에 랭크되었으나 시간이 가면서 유저의 이탈이 많아진 게임이다. 본 게임의 데이터는 'Microsoft gaming data hackaton' 을 통해 공개되어있다[9].

그림 8 게임화면: Delivery outlaw

세번째 게임은 'Tagpro'이다. 이 게임은 온라인 캐주얼 CTF(capture the flag) 게임이다[그림 9]. 이 게임은 나머지 게임과 다르게 다른 유저들과 플레이가 가능하다. 또한, 간단한 조작으로 유저들과 협력 및 경쟁을 할 수 있다. 본 게임에서 사용된 데이터는 Tagpro 자체에서 제공한 Open api를 이용해서 가져올 수 있다[10].

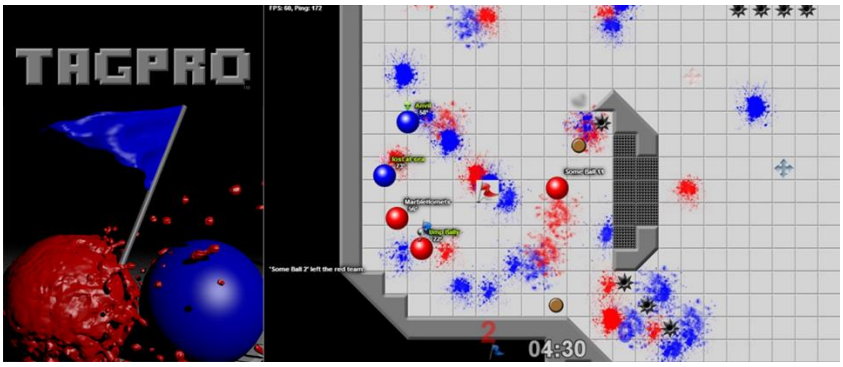

그림 9 게임화면: Tagpro

본 연구에서는 총 3종의 캐주얼 게임의 데이터를 사용하여 유저 이탈 예측 연구를 하였다. 그리고 게임 내 유저 데이터를 이용하여 이탈 예측을 위한 모델을 설계하고 다양한 관점에서 예측 성능에 끼치는 영향에 대해서 연구하였다. 또한, 유저 이탈 예측 연구를 하는 과정에서 서로 다른 3개 게임의 공통점과 차이점을 비교 분석하면서 어떤 차이가 발생하는지 알아보도록 하였다.

제 2 절 연구의 필요성

대부분의 캐주얼 게임의 수익은 부분유료화 방식을 취하고 있다. 부분유료화란 기본 게임은 무료로 제공해주고 업그레이드, 아이템, 추가 기능 해제, 진행속도 등을 앱내 결제로 구매하도록 하는 비즈니스 모델이다[8]. 또한, 여러 광고를 게임 내에 노출 시키는 방법으로 추가적인 수익을 낼 수도 있는데, Admob, Tapjoy 등의 광고 플랫폼을 통해서 광고주가 요청한 광고를 게임 내에 삽입하여 유저가 보거나 클릭할 때 수익이 나도록 하는 모델이다. 최근 모바일 캐주얼 게임은 주로 이 두 가지 방법을 섞어서 사용하고 있다[11].

한국콘텐츠진흥원의 보고서에 따르면 이런 부분유료화 방식을 사용하는 게임을 플레이하는 유저 중 한번이라도 결제해본 경험이 있는 유료 유저의 10%가 전체 게임 매출의 46.4%를 차지한다고 한다. 또한, 유료 유저들이 처음으로 게임 아이템을 구입하는 기간은 게임 시작 후 일주일이 54.9%로 가장 많이 나타났다[그림 10][12].

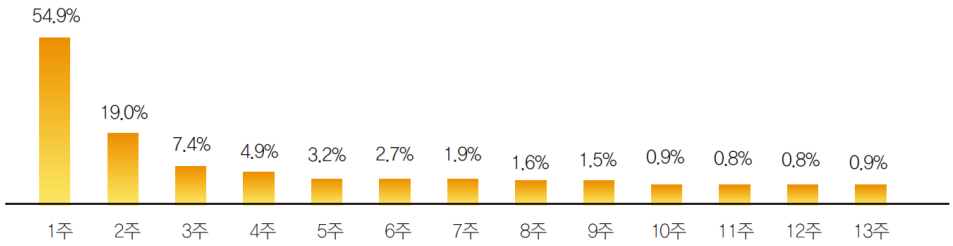

그림 10 유료 유저들의 기간별 아이템 구입 비중

[그림 10]에서 설명한 부분유료화 게임을 이용하는 유료 유저들의 기간별 아이템 구입 비중을 본다면 유저 한 명이 이탈하는 것이 매출과 바로 직결될 수 있음을 생각해 볼 수 있다. 또한, 이탈하는 유저를 예측하고 해당 유저에게 게임 내외적으로 추가적인 이익을 주어 이탈을 방지한다면 추가적인 수익이 가능하다. 즉, 이탈 유저를 사전에 예측하면 이탈률이 감소되고 이는 매출 상승에 직접적인 영향을 주게 된다[13].

유저 이탈 예측을 통한 데이터 분석을 한다면 게임 전반의 기획,

개발, 마케팅 전략에 데이터 기반의 Insight 제공이 가능하다[14]. 이는 유저 이탈에 대한 정량적인 분석으로 도출한 Insight를 통해 게임의 난이도, 배점 시스템, 보상 시스템 등의 게임 매커니즘을 기획하고 개발하는 것을 포함한다.

유저 이탈 예측 모델은 모바일 캐주얼 게임뿐만 아니라 MMORPG (Massive Multiplayer Online Role Playing Game)와 같은 게임에서도 중요한 역할을 한다. World of Warcraft를 예를 들 수 있는데, 이 게임은 유료게임으로 한 달마다 일정 비용을 받는 비즈니스모델을 가지고 있다[15]. 한 달마다 일정 비용을 받는 비즈니스모델이기 때문에 유저 이탈에 더욱 더 민감하고 한 명의 유저를 놓치는 것이 한 달의 비용이 아닌 일 년의 비용을 손해 보는 것으로 본다. 위와 같은 비즈니스 모델에서는 유저 이탈 예측모델이 매우 중요하고 이탈률을 감소시키는 것이 매출과 직결된다.

유저의 이탈 예측의 중요성은 이탈 예측을 통한 유저의 유지 비용과도 연관된다. 신규 유저의 유입 비용은 기존 유저의 유지 비용보다 5배가 더 많다[16]. 즉, 신규 유저를 한 명 유입시키는데 드는 비용이 기존 유저 5명을 유지시키는데 비용과 동일하다는 것인데. 이는 유저 이탈 예측을 통해서 나갈 유저를 유지하는 데 드는 비용으로도 환산할 수 있다.

그러므로 유저 이탈 예측 연구는 매출 상승에 직결되는 뿐만 아니라 객관적인 데이터 기반의 Insight를 제공하게 되며, 이에 본 연구의 중요성이 있다고 할 수 있다.

제 3 절 연구의 목적

앞서 설명하였듯이 유저 이탈 예측은 게임 산업에서 많은 중요성을 가지고 있다. 유저 이탈 예측 모델을 통해 이탈이 예측되는 유저들을 사전에 예측하여 이탈을 막는 효과를 줌으로써 유저의 유지 면에 큰 기여를 할 수 있다. 또한, 정량적인 데이터 분석을 통해 새로운 Insight 발견이 가능하다. 연구의 목적은 아래와 같이 세가지로 정한다.

첫째, 캐주얼 게임에서의 유저 이탈 예측 모델을 제안하고자 한다. 모바일 캐주얼 게임 2종과 온라인 캐주얼 게임 1종의 데이터를 가지고 데이터 분석과정을 통하여 유저 이탈 예측 모델을 설계하고, 모델별로 실험한 후, 성능 평가를 통해 예측 모델을 제안한다.

둘째, 유저 이탈 예측에 대한 기존 연구의 한계를 개선하고자 한다. 앞으로 다룰 게임 분야의 유저 이탈 예측 관련한 연구들의 한계점을 찾아보고 해당 한계점을 개선할 수 있는 부분을 찾아본 연구에 적용하도록 한다.

셋째, 이탈 예측 분석 연구를 통한 새로운 Insight를 발견하고자 한다. 유저 이탈 예측 분석 방법의 단계별로 데이터 분석을 하고 Domain Knowledge를 활용한 시각을 통해 새로운 Insight를 발견하려고 한다. 또한, 일련의 연구 과정을 통해서 게임 장르와 플랫폼의 차이가 가져다 주는 Insight에 대해서도 알아보기로 한다.

제 2 장 관련 연구

제 1 절 게임 데이터 분석 연구

최근 들어 게임의 수요가 급증하고 게임 산업이 커짐에 따라 게임을 연구적인 관점에서 보는 노력이 다양하게 이루어지고 있다. 게임에 대한 연구는 게임의 메커니즘을 보거나 게임의 인터페이스에 대한 분석 등 다방면에 걸쳐서 이루어지고 있다[17, 18]. 특히 게임 안에 있는 여러 게임 데이터들을 활용하여 게임 내의 현상을 분석하는 연구들도 많이 이루어지고 있다.

Feng et al. (2007)은 MMORPG(Massively Multi-player Online Role Playing Game)분야인 'EVE Online'의 5년 동안의 유저 세션 데이터를 가지고 연구를 진행하였다[19]. 트래픽 데이터, 인구 데이터, 개인 데이터 총 3가지 분야로 연구를 진행하였으며, 트래픽 데이터를 통해서 일주일 동안의 규칙적인 패턴 변화에 관해서 설명하였다[그림 11].

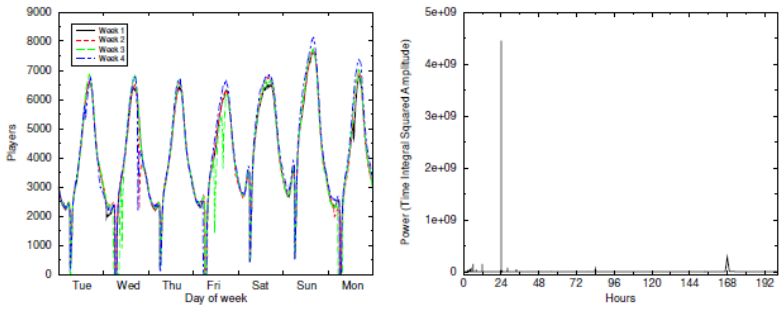

그림 11 (좌)한달 간의 데이터(일주일 단위), (우) 일주일 데이터의 FFT

인구 데이터를 통해서 업데이트 시기에 따른 인구의 증감 현상을 설명하였다. 또한, 개인 데이터들의 분석을 통해서 게임 초반, 중반, 후반 시기에 따른 플레이 시간의 차이를 설명하였다.

Bauckhage et al. (2012)는 총 20만 명이 넘는 유저들의 총 5개 유명 게임(Battle Field Bad Company 2 (BF2), Crysis2 (CR2), Medal of Honor (MOH), Just Cause 2 (JC2), and Tomb Raider: Underworld (TRU))에 대한 게임 플레이 시간의 데이터를 가지고 시간의 분포를 총 4개의 분포(Weibull fit, Gamma fit, Log−normal fit, Gaussian fit)에 매칭시켜서 살펴보았다[20] [그림 12].

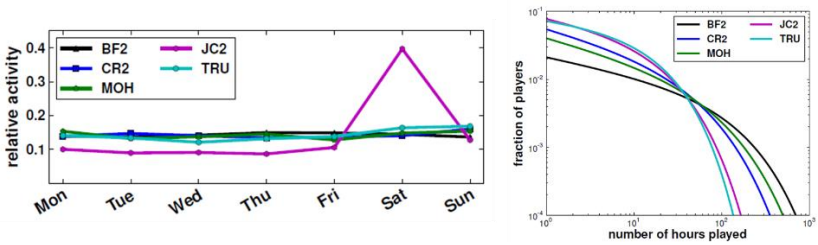

그림 12 5개 게임의 평균 플레이 시간 비교

게임 내의 데이터뿐만 아니라 게임 외의 데이터를 가지고 비교하는 연구도 이루어졌다. Williams et al. (2008)은 EverQuest2 유저들을 대상으로 게임 안에서 실제 설문을 하여 데이터를 수집하였다[표 1].

Educational level	Percentage of EQ2 players	Percentage of General Population
Less than high school	7.67	20.14
High school diploma	15.62	29.82
Some college	32.63	18.21
Associates degree	16.93	7.78
Bachelor's degree	14.43	16.01
Grad training or prof degree	12.67	8.03

표 1 EverQuest2 유저들의 학력과 미국 평균 학력 비교

게임 유저의 실제 나이, 성별, 인종, 수입, 교육, 종교, 비만 지수, 미디어 사용 등의 실제 오프라인 정보들을 모아서 실제 미국 평균치와 비교 분석하였다[21].

제 2 절 유저 이탈 예측 연구

초기의 유저 이탈 예측 연구는 다양한 분야에서 이루어져 왔다. 통신사 고객의 이탈에 관한 연구, 신문 구독자들의 이탈에 관한 연구, 은행 고객의 이탈에 관한 연구, 신용카드 고개의 이탈의 관한 연구, 보험 고객의 이탈에 관한 연구들이 있었다[23-26]. 위와 같이 유저 이탈에 관한 연구는 각광받는 산업이 옮겨감에 따라 연구 분야도 계속 변하면서 이루어져 왔다.

최근 게임 산업이 각광받음에 따라, 게임 분야의 유저 이탈은 게임의 수익과 직결되는 문제이기 때문에 유저의 이탈을 예측하기 위한 몇몇 연구들이 이루어졌다[8].

Runge at al(2014)는 유저 이탈 예측 연구를 비즈니스적인 관점에서 풀어냈다[27]. 2가지의 부분유료화 모바일 캐주얼 게임의 유저를 중 결제한 비용이 많은 순으로 상위 10%의 유저들을 대상으로 유저 이탈을 예측 하였다[그림 13].

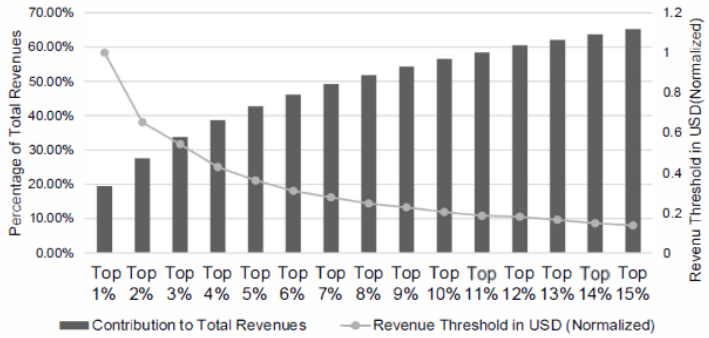

그림 13 전체 매출 대에서 매출 상위 유저들의 매출 기여도

게임 내의 데이터들을 분석하여 Feature를 추출하고 해당 Feature들을 가지고 데이터 변환을 한 후에 4가지 알고리즘(뉴럴 네트웍스, 로지스틱 회귀, 의사결정나무, 서포트벡터머신)을 통해서 만든 예측 모델에 적용하였다. 만들어진 모델은 ROC곡선의 AUC(Area under Curve)를 가지고 성능 평가를 하였다. 성능이 가장 좋게 나온

모델을 통해 실제 유저에게 얼마나 영향을 끼치는지 알아보기 위해 A/B테스트로 실험군과 대조군을 나누고 유저 이탈 예측 모델을 통해 예측한 유저와 실제 이탈한 지 얼마 안 되는 유저들에게 알람을 주고 게임의 복귀율을 테스트해본 결과, 예측모델을 사용하여 이탈 예정인 유저의 복귀율이 더 높게 나타났다.

Hadiji et al(2014)는 유저 이탈에 대한 정의를 2가지 개념으로 나누어서 설명하였고, 유저 이탈 예측에 사용할 Input Data 생성도 2가지 방법으로 설명하였다[14] [그림 14].

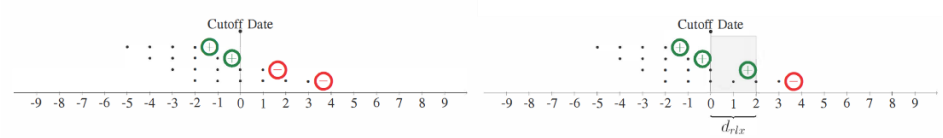

그림 14 두 가지의 유저 이탈의 정의

부분유료화 게임의 데이터를 가지고 2가지의 유저 이탈에 대한 정의와 Input Data 생성에 대한 2가지 방법으로 생성한 4가지 Input Data를 4가지 예측 모델(의사결정나무, 로지스틱 회귀, 뉴럴 네트워크, 나이브베이즈)을 통해 모델링 하였다. 성능은 F1-Score로 비교하였으며 의사결정나무를 사용한 예측 모델이 성능이 가장 좋은 것으로 나타났다.

Kawale et al(2009)은 MMPROG의 유저 데이터를 통해서 유저 이탈에 영향을 끼치는 요소를 사회적인 요소와 몰입적 요소로 살펴보았다[28] [그림 15].

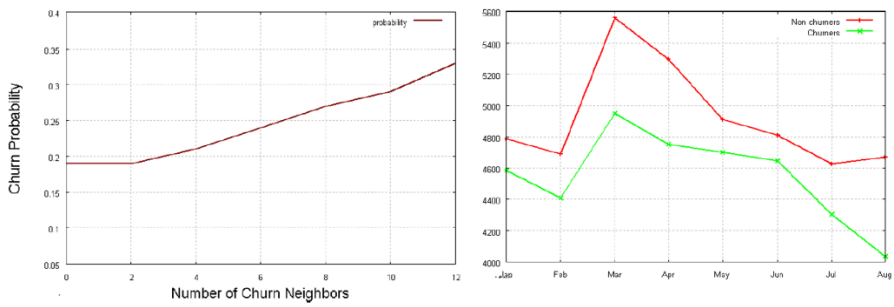

그림 15 (좌)이탈하는 이웃 수에 따른 이탈률, (우)월 별 이탈유저와 이탈 안 한 유저

사회적인 요소는 그룹플레이를 한 기록들을 가지고 Network-graph를 생성하여 분석하였고 몰입적인 요소는 유저의 Session 시간을 통해서 Beta-function으로 살펴보았다. 그리고 기존의 Simple Diffusion Model을 수정하여 Modified Diffusion Model을 만들어 성능 평가를 하였고 이탈하는 유저와 이탈하지 않는 유저를 제작한 모델로 분류하는 과정을 설명하였다.

최근 게임 분야의 유저이탈 예측에 대한 중요성이 부각되며 관련 연구가 이루어지고 있지만 아직까지는 많이 부족한 상황이다. 앞서 설명한 기존의 유저 이탈에 대한 연구도 세가지 한계점을 가지고 있다.

첫 번째로 이탈 예측을 하기 위해서 Feature를 정의하는 데 대부분의 연구가 Feature에 대해 정의만 할 뿐 Feature 각각에 대한 깊이 있는 접근이 없다. Feature를 어떻게 정의하느냐에 따라서 예측 성능이 달라질 수 있는데 Domain Knowledge를 활용한 Feature의 심도 있는 접근을 하지 않는다면 예측모델의 성능 향상을 기대하긴 어렵다[29]. 이에 본 연구에서는 Feature를 정의함과 동시에 각 Feature 별로 깊이 있는 접근을 통해서 예측 성능 향상을 이루고자 한다.

두번째로 기존 게임의 이탈 예측 연구에서는 관찰 범위와 이탈 예측 범위 설정에 추가적인 연구 없이 임의로 정하여 이탈 예측 연구를 하고 있다. 그러나 이탈 예측 연구에서는 관찰 범위와 이탈 예측 범위에 따라서 예측 결과 및 성능은 많이 달라진다. 이러한 한계점을 통해서 본 연구는 관찰 범위와 이탈예측 범위의 변화에 따라서 예측 성능에 미치는 영향에 대해서 알아보고, 관련 유저 이탈 예측 연구에 도움이 되고자 한다.

마지막으로 이탈 예측 모델을 설계하는데 있어서 Feature 뿐만 아니라 예측 모델에서 쓰이는 알고리즘도 예측 성능에 큰 영향을 미친다. 그러나, 기존 게임 이탈 예측 연구에서는 예측 모델에서 쓰이는 알고리즘에 대해 제한적인 접근을 하고 있다. 본 연구에서는 세가지 대표적인 알고리즘을 가지고 예측 모델을 설계하며 예측 결과를 알고리즘 별로 비교 분석하기로 한다.

제 3 장 연구 문제 및 방법

제 1 절 연구 문제

본 연구에서는 2종의 모바일 캐주얼 게임과 1종의 온라인 캐주얼 게임의 유저 데이터를 가지고 유저 이탈 예측 모델을 설계한 후, 모델의 성능 평가 결과를 살펴보도록 한다. 일련의 과정 동안 Feature의 개별적인 분석과 예측 성능에 끼치는 영향을 알아보돌고 한다. 또한, 관찰 기간 및 이탈예측 기간에 따른 예측 성능에 대에서 살펴보고 대표적인 3개의 알고리즘 별 예측 성능에 대한 분석을 한다. 아래에서 설정한 세가지 연구문제는 연구의 목적과 기존 관련 연구의 한계점을 반영하여 설정하였다.

연구문제 1. 유저 이탈 예측 모델에 있어서 Feature들이 예측 성능 향상에 얼마나 기여하는가?

연구문제 1에서는 유저 이탈 예측 성능에 크게 기여하는 Feature를 살펴보고자 한다. 이를 위하여 각 게임별 Feature들을 정의한다. 공통으로 정의되는 Feature와 한 게임에 종속적으로 정의되는 Feature들을 가지고 다양한 관점에서 살펴보고 이 Feature들을 가지고 유저 이탈 예측을 하였을 때 성능 향상에 얼마나 기여하였는지 알아보도록 한다.

연구문제 2. 유저 이탈 예측에 있어서 관찰 기간 및 이탈예측 기간은 예측 성능에 어떠한 영향을 미치는가?

연구문제 2에서는 유저 이탈 예측을 함에 있어서 관찰 기간과 이탈예측 기간이 이탈 예측 결과에 얼마나 영향을 끼치는지 알아보도록 한다. 총 3종 게임의 관찰 기간과 이탈예측 기간을 일정 범위를 정해둔 채로 각각 조절해가면서 기간 변화에 따른 이탈 예측 성능을 분석해보기로 한다.

연구문제 3. 유저 이탈 예측 모델에 있어서 알고리즘 별 예측 성능은 어떠한가?

연구문제 3에서는 이탈 예측 모델의 알고리즘 별로 예측 성능에 끼치는 영향에 대해서 살펴보고자 한다. 총 3개의 알고리즘(Gradient boosting, Logistic regression, Random forest)을 통해서 유저 이탈 예측 모델을 제작하고 성능평가 결과에 대해서 비교해보고 각 게임 별로 예측 성능 결과에서 나타나는 공통점과 차이점에 대해서도 알아보도록 한다.

제 2 절 연구 방법

유저 이탈 예측 연구의 연구 방법은 기존의 데이터 분석을 통한 예측 방법과 유사한 방법을 사용하도록 한다. 본 연구에서의 흐름을 도식화하면 [그림 16]와 같다

그림 16 연구 흐름도

데이터 전처리 단계에서는 게임 별로 전처리 단계가 차이가 있기 때문에 각각 설명을 하기로 한다. 첫번째 게임(Crap

dodging)에서는 데이터가 서버에 저장되는 방식을 설명하고 서버에 있는 원본데이터를 가져오는 방식에 대해서 설명한다. 두번째 게임(Delivery outlaw)과 세번째게임(Tagpro)은 데이터를 가져온 방식과 형태에 대해서 설명한다. 또한, 데이터 처리를 위해서 미리 데이터를 전처리하는 것에 대해서도 설명 한다.

Feature 정의 단계에서는 3게임 공통적으로 적용할 Feature 10가지(playCount, bestScore, consecutivePlayRatio, worstScore, meanScore, sdScore, bestSubMeanRatio, bestSubMeanCount, bestScoreIndex, activeDuration)와 첫번째 게임을 제외한 나머지 2개의 게임별로 종속적으로 적용할 4개의 Feature들(purchaseCount, bestPurchase, winRatio, gameDurationMean)을 정의한다.

Feature 정의단계가 끝난 후 예측 모델을 설계하도록한다. 예측 모델별로 알고리즘을 정하는 데 본 연구에서는 Gradient boosting, Logirstic Regression, Random Forest 알고리즘을 통해서 예측 모델을 설계하도록 한다.

데이터 변환 단계에서는 앞서 설계한 유저 이탈 예측 모델의 Input데이터로 사용할 수 있게 변환 과정을 거친다. 전처리 단계에서 가공한 데이터를 가지고 각 Feature 별로 테이블 변환과정을 거친 후에 예측 모델에 Input데이터로 사용할 수 있는 통합 테이블로 변환시킨다.

모델별 실험 및 분석 단계에서는 전 단계에서 변환된 데이터로 앞서 설계한 세 가지 유저 이탈 예측 모델(Gradient boosting, Logirstic regression, Random forest)에 Input으로 넣어 Output결과를 살펴본다. Output결과는 ROC(Receiver-operating characteristic) Curve를 통해서 살펴보도록 한다.

성능 평가 및 비교 단계에서는 전 단계에서 설계한 모델을 가지고 10-fold cross validation을 통해서 ROC를 구하고 모델별, Feature별, 게임 별 AUC(Area under the curve)를 비교하면서 성능 평가를 하도록 한다.

제 4 장 유저 이탈 예측 모델 설계

4장에서는 본 연구에서 제안하고자 하는 유저 이탈 예측 모델의 설계 방법 및 과정에 관해 서술하고자 한다. 1절에서는 예측 모델을 설계하기 전에 관찰 기간(Obsercation period)과 이탈예측 기간(Churn prediction period)에 대해서 설명을 하고 유저 이탈을 정의하도록 한다. 2절에서는 각 게임 별로 데이터 전처리(Data-preprocessing)에 대해서 설명하도록 한다. 첫번째 게임(Crap dodging, 이하 Game 1)은 원본데이터를 생성하는 방식과 원본 데이터의 구조 그리고 데이터 전처리 과정에 대해서 설명하기로 한다. 두번째 게임(Delivery outlaw, 이하 Game 2)과 세번째 게임(Tagpro, 이하 Game 3)는 원본데이터 구조에 대한 설명과 전처리 과정에 대해서 설명을 한다. 3절에서는 3개의 게임이 공통되게 적용할 수 있는 10가지 Feature와 Game 2, Game 3 각각 종속적으로 적용가능한 2가지(총 4가지) feature를 정의(Feature Definition)하고 해당 Feature 별로 기본 통계를 보도록 한다. 4절에서는 본 연구에서 제안하는 유저 이탈에 대한 예측 모델을 각각의 알고리즘(Gradient boosting, Logistic regression, Random forest) 별로 설계하도록한다.

제 1 절 유저 이탈에 대한 정의

유저 이탈 연구를 하기 위해서는 유저 이탈에 대한 명확한 정의가 있어야 한다. 또한, 유저 이탈 예측 모델을 만들기 위해서는 데이터의 명확한 범위가 설정되어야 한다. 1절에서는 유저 이탈을 정의하고 이탈 예측 모델을 설계하기 위해 관찰 기간(Observation period)와 이탈예측 기간(Churn prediction period)에 대해 설명을 하고 유저 이탈에 대한 정의를 하도록 한다.

1. 관찰 기간(Observation period)

본 연구에서 관찰 기간이란 이탈 예측의 기준이 되는 기간이다.

즉, 이탈 예측을 하는 근거가 되는 데이터의 범위를 정하는 기간을 관찰 기간이라고 말한다. 이 관찰 기간은 데이터의 성격이나 예측 의도 등에 따라서 바뀔 수 있다. 그러나 하나의 유저 이탈 예측 모델 내에서는 하나의 관찰 기간 범위만 존재한다. 이탈 예측을 위한 관찰 기간을 설정하기 위해서는 기본적으로 분석 대상 유저들의 시간을 기반으로 한 플레이 기록이 필요하다. 유저 별로 시간을 기반으로 한 플레이 기록이 정리가 되면, 각 유저 별로 첫 플레이 시각을 0으로 만들어 나열해두어 관찰 기간 설정이 가능한 상태로 만들어 놓는다. 아래 [그림 17]는 관찰 기간에 대한 설정을 도식화하여 설명하고 있다.

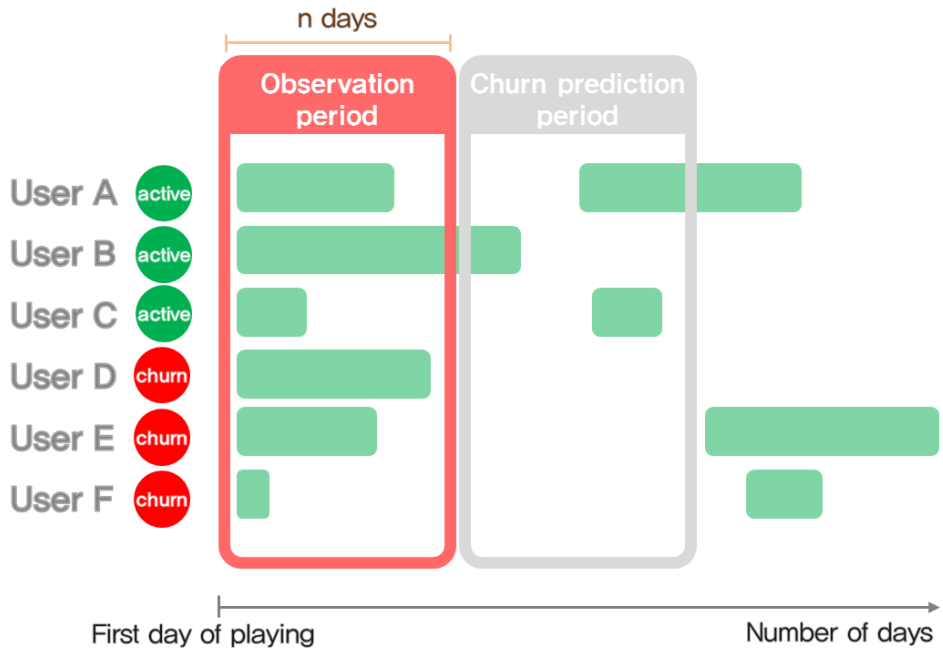

그림 17 관찰 기간 설정 예시

위에서 User A부터 F까지 모든 유저가 첫 플레이시각을 0으로 맞추어 나열되어있다. 그리고 유저의 첫 플레이시각부터 일정 기간까지를 관찰 기간으로 정해두고 해당 관찰 기간에 속해있는 유저들의 데이터를 가지고 유저 이탈 예측을 위한 Feature를 추출하도록 한다. 관찰 기간내에 유저의 플레이기록이 없는 경우는 플레이를 전혀 하지 않은 유저이며, 이 유저의 경우에는 분석에서 제외하도록 한다.

2. 이탈(Churn)과 이탈예측 기간(Churn prediction period)

본 연구에서 유저 이탈이라고 함은 유저가 더 이상 게임을 하지 않는 것을 의미하며 유저 별로 플레이 데이터를 통하여 유저가 이탈하였는지 이탈하지 않았는지 분류 가능하다. 본 연구에서는 유저가 이탈하지 않았으면 활성(active)유저라 명한다.

이탈예측 기간이란 유저의 이탈 여부를 판단하는 기간이다. 앞서 다루었던 관찰 기간(Observation period)이 끝나는 바로 다음기간부터 일정 기간을 설정하여 이탈예측 기간을 잡는다. 이탈예측 기간은 관찰 기간과 마찬가지로 데이터의 성격이나 예측 의도에 따라서 유동적으로 기간의 범위 설정이 가능하다. 이탈예측 기간도 마찬가지로 하나의 유저 이탈 예측 모델 내에서는 하나의 이탈예측 범위만 존재한다. 유저의 이탈은 해당 이탈예측 기간동안 유저의 플레이 데이터가 없으면 유저 이탈로 분류하고 해당 이탈예측 기간동안 플레이 데이터가 존재하면 활성(active) 유저로 분류한다. 유저 이탈을 분류하는 과정은 [그림 18]에 도식화하여 나타내었다.

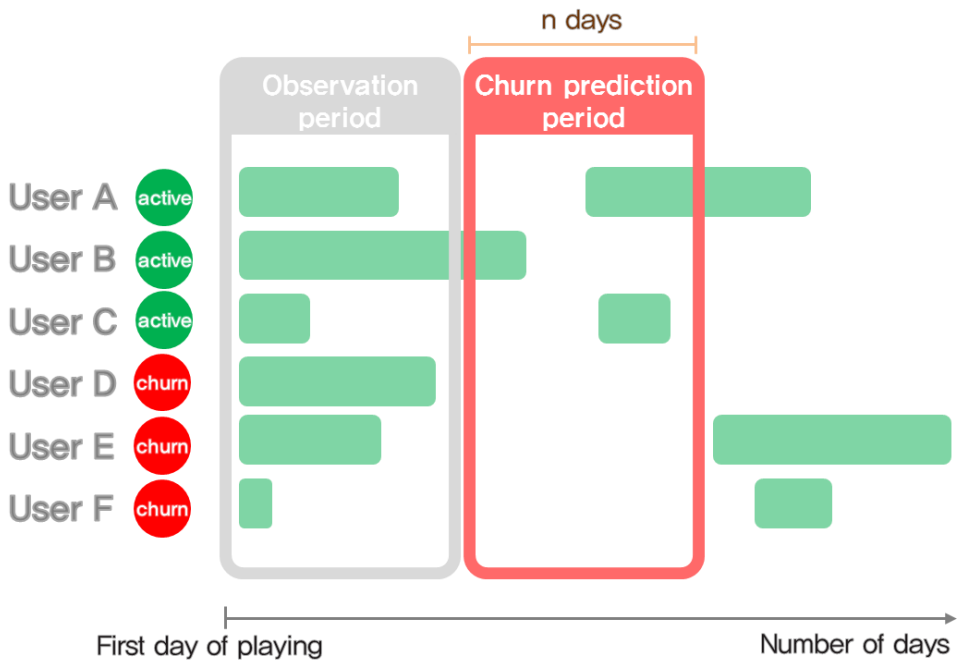

그림 18 유저 이탈 분류 과정

앞서 설명한거와 같이 이탈예측 기간 범위 안에 유저의 플레이 존재 여부에 따라 이탈유저와 활성유저가 나뉘어진다. User A는 관찰 기간 이후 이탈예측 기간에 플레이 기록이 존재하므로 활성유저이다. User B와 User C도 마찬가지로 이탈예측 기간에 플레이 기록이 존재하므로 활성유저로 분류가 된다. 그러나 User D의 경우엔 관찰 기간 이후로 플레이 기록이 없기 때문에 이탈 유저로 분류가된다. User E와 User F의 경우에는 관찰 기간 이후 플레이 기록이 존재를 한다. 그러나 이탈예측 기간 범위 안에 플레이 기록이 존재하지 않기 때문에 이탈 유저로 분류가 된다. 만약 이탈 예측 연구의 상황에 따라서 User E와 User F를 활성유저로 분류를 해야 한다면, 관찰 기간을 늘리는 방법 혹은 이탈예측 기간을 늘리는 방법으로 유저의 이탈 여부 변환이 가능하다.

앞서 유저 이탈과 관찰 기간에 대해서 정의한 것을 토대로 3개 게임 데이터의 모든 유저들을 가지고 관찰 기간에 따른 유저 이탈률의 변화량 그래프를 [그림 19]과 같이 표현하였다.

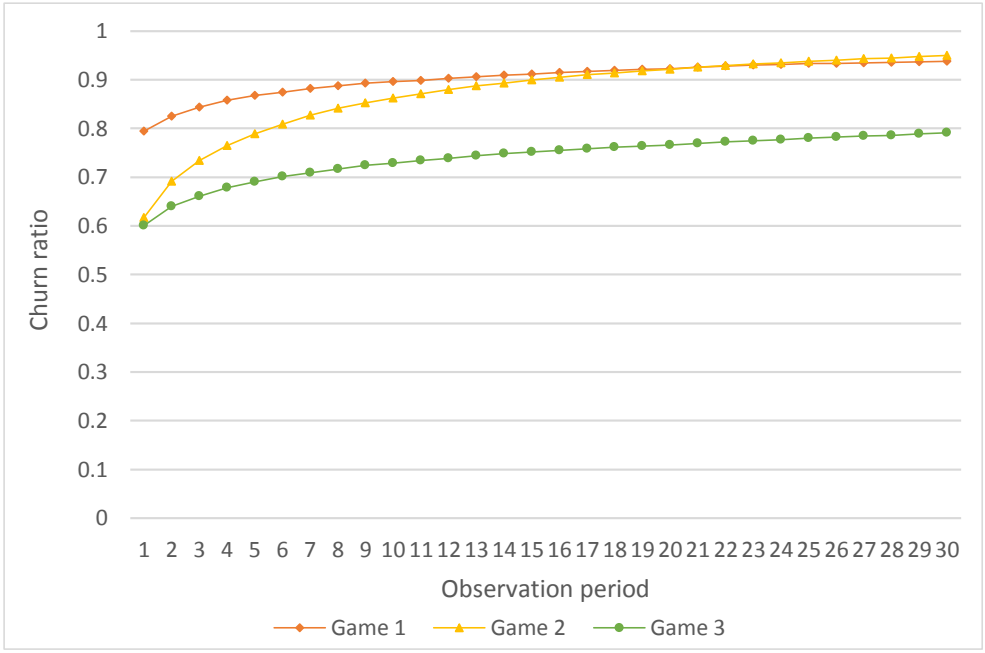

그림 19 관찰 기간 별 유저 이탈률

위의 그래프를 보면 Game 1의 경우는 하루만에 이미 약 80%의 유저가 이탈을 하였고 10일 이후에는 약 90%의 유저들이 이탈을

끝마친 상태이다. 그리고 그 이후 기간동안 이탈률의 큰 변화 없이 수렴되는 곡선을 그리고 있다. Game 2의 경우엔 Game 1에 비해 하루 동안은 약 60%로 적은 이탈률을 보여주고 있으나 관찰 기간이 길어짐에 따라서 급격하게 이탈률이 오르고 20일 이후에는 Game 1과 비슷한 수치가되어 Game 1과 마찬가지로 수렴되는 곡선을 그리고 있다. Game 3의 경우 첫 하루동안에는 Game 2와 같이 60%의 이탈률을 보였으나 그 이후의 유저의 이탈 속도는 Game 2에 비해 낮게 나타났다. 또한, 관찰기간이 30일이 되어도 약 95%의 이탈률을 보여주고 있는 Game 1과 Game2에 비해 Game 3는 80%의 이탈률로 낮은 이탈률을 보여주고 있다. 이는 모바일 플랫폼인 Game 1과 Game 2가 온라인 플랫폼인 Game 3에 비해 접근성이 좋은만큼 이탈할 수 있는 기회도 온라인 플랫폼에 비해 많다는 해석이 가능하다. 그리고 세개의 게임 공통으로 일주일동안의 이탈률의 변화량만 보아도 앞으로 그래프가 어떤 형태로 진행이 될지 예측이 가능하다고 볼 수 있다. 이렇게 관찰기간 별 이탈률의 변화량만을 보아도 서비스 하고 있는 게임에 대해서 차후 플랜의 방향을 잡는데 도움 될 수 있다.

제 2 절 데이터 전처리(Data-preprocessing)

본 연구에서 사용되는 게임 데이터는 총 3개의 게임에서 추출한 데이터이며, [표 2]은 각 게임별 데이터에 대한 요약이다.

분류	Game 1	Game 2	Game 3
게임 제목	Crap dodging	Delivery outlaw	Tagpro
장르	캐주얼 액션 게임	캐주얼 레이싱 게임	캐주얼 CTF 게임
플랫폼	Android	Android, iOS	Web
데이터 기간	2015-01-15 ~ 2016-05-24 (약 13개월)	2014-06-16 ~ 2014-10-01 (약 3개월)	2015-05-26 ~ 2015-12-25 (약 7개월)
데이터 수	153,876개	7,620,127개	3,100,955개
유저 수	25,954명	79,436명	88,043명

표 2 게임 별 데이터 정보

본 연구에서 수집한 데이터들을 가지고 유저 이탈 예측에 사용하기 위해서는 가공 없이 원본 데이터를 그대로 사용할 수 없다. 여러 분석 툴을 사용해서 돌려야 하는 데이터 분석 특성상 불완전한 원본 데이터를 가지고 분석을 하는 데 무리가 있다. 그러므로 데이터 전처리 과정을 통해 데이터를 완벽하게 가공하도록 한다.

1. 원본 데이터 수집

Game 1의 데이터는 Googleplay에 2011년부터 서비스가 되고 있는 게임의 유저 데이터이다. Game 1의 데이터는 한 게임이 끝나고 유저의 점수가 php로 제작한 api를 통해서 서버에 있는 데이터베이스 안의 테이블로 저장된다. [그림 20]에 위 과정이 도식화되어 있다.

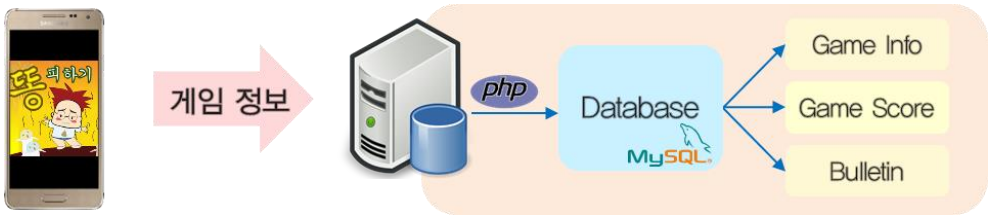

그림 20 데이터 수집 방법

Game 2의 데이터는 'Microsoft gaming data hackaton'을 통해 공개된 게임 데이터이고, Game 3의 데이터는 Tagpro 자체에서 제공하는 Api를 통해서 수집가능하다.

2. 데이터 전처리

Game 1

유저 이탈 예측에 사용할 데이터는 MySQL 데이터 베이스에 저장되어 있다. MySQL에 있는 데이터는 쿼리를 통해서 추출 할 수 있고 myPhpAdmin을 통해서 웹 페이지에서도 실시간으로 손쉽게 확인이 가능하다. 유저가 게임 데이터를 저장하면 데이터베이스에 있는 유저 테이블에 하나의 열로 저장이 된다[표 3]. 또한, 유저테이블에 있는 속성은 총 5개로 [표 4]에 설명이 되어있다.

id	device	score	time	totaltime
109688	3558270524154	0	2015-10-29 00:10:12	4
109689	3515100603747	288	2015-10-29 00:36:12	95
109690	3538760603990	5	2015-10-29 06:31:20	6
109691	3575640605869	18	2015-10-29 07:55:01	22
109692	3575640605869	149	2015-10-29 07:56:12	61
109693	3575380659697	421	2015-10-29 08:17:41	81
109694	3575380679568	615	2015-10-29 09:17:46	111
109695	3558270546568	433	2015-10-29 09:48:43	72
109696	3558270546568	797	2015-10-29 09:50:57	108
109697	3576300515657	7	2015-10-29 09:55:24	16
109698	3579410620834	255	2015-10-29 11:10:38	73

표 3 데이터 저장 형태

속성	내용
id	등록순으로 증가되는 데이터 고유 id
device	디바이스 별 부여되는 고유 id
score	유저가 게임에서 기록한 점수
time	점수 등록 시 서버의 시간
totaltime	점수 등록 한 게임 플레이 시간(초단위)

표 4 데이터의 속성 설명

연구에 사용할 데이터에 대한 분석 및 모델링은 R을 사용한다. R에서 데이터 분석으로 사용하기 위해서는 MySQL에 있는 데이터를 csv파일로 변환하여 R에 import한다. 속성 중 time부분은 MySQL에서 '년월일시간' 포맷으로 표현되고 있다. 표현된 시간 형식은 R을 통해서 분석하기에는 빠른 분석이 되지 않는다. 예를 들어 R에서 '년월일시간' 의 데이터 포맷이 다른 시간과의 연산을 하기로 하면 csv에서 해당 포맷을 문자열포맷으로 읽어와 시간포맷으로 형변환이 되어 연산이 된다. 위 포맷을 가지고 데이터가 많은 파일에서 연산한다고 하면 많은 시간적 낭비가 일어난다. 위 포맷을 MySQL에서 쿼리를 통해 Unix 시간 포맷으로 변환 되면 숫자 형태기 때문에 R에서 최적화된 연산이 가능하다[30]. 위 의 시간관련 속성의 전처리를 위한 쿼리는 [표 5]과 같다.

Input	2015-05-24 11:53:23
Query	SELECT UNIX_TIMESTAMP(time)
Output	1432468403

표 5 전처리 쿼리: time 속성

최종적으로 MySQL에서 유저 데이터 파일을 추출하여 csv파일로 저장한다. 저장된 csv파일은 R에서 직접 Load가 가능하다. 최종적으로 전처리된 데이터 테이블은 device, score, time으로만 이루어져 있는 테이블로 완성이 된다.

Game 2

Game 2에서 제공하는 데이터 형태는 JSON 포멧으로 제공해주고 있다. 본 연구에서는 2가지의 데이터 구조를 사용하고 있으며 2가지 데이터의 JSON구조가 다르다. [그림 21]은 하나의 게임 플레이를 하였을 때 서버에 보내는 데이터에 대한 구조를 보여주고 있고 [그림 22]는 상점에서 구매를 하였을 때 서버에 보내는 데이터의 구조이다. 또한, [표 6]은 Game 2의 주요 데이터의 속성에 대한 설명이다.


```
{
  "time":1406267268241,
   "date":"2014-07-25T05:47:48.241Z",
   "clientIp":"xxx.xxx.xxx.xxx",
   "app":"DO",
   "appVersion":"2.4.67.7442",
   "device":"iPod",
   "event":"progress",
   "localTime":"1406267236000",
   "platform":"iOS",
   "properties":{
      "action":"complete",
      "build":{
         "chassis":{
            "level":1,
            "upgradelevel":1
         },
         "engine":{
            "level":1,
            "upgradelevel":2
         },
         "gearbox":{
            "level":1,
            "upgradelevel":2
         …(truncated)
      },
      "id":"junkyard_2",
      "package":98,
      "reward":303,
      "type":"race"
   },
   "queueDuration":"29000",
  "uid":"D9ECB34E9A1D31DE15B334E32001B32BD",
   "calcDate":"2014-07-25T05:47:19.0000000"
},
```


그림 21 Game 2 게임 플레이 데이터 구조

```
{
    "time":1406267278630,
    "date":"2014-07-25T05:47:58.63Z",
    "clientIp":"xxx.xxx.xxx.xxx",
    "app":"DO",
    "appVersion":"2.4.67.7442",
    "device":"iPod",
    "event":"softPurchase",
    "localTime":"1406267271000",
    "platform":"iOS",
    "properties":{
        "balance":2465,
        "currency":"soft",
        "id":"scooter_clone_suspension_1_2",
        "price":400,
        "remaining":2065,
        "type":"vehicle_upgrade"
    },
    "queueDuration":"4000",
    "uid":"0001E7EDE15X33XE32001B32BX",
    "calcDate":"2014-07-25T05:47:54.0000000"
},
```


그림 22 Game 2 상점 이용시 데이터 구조

속성	내용
time	데이터가 생성된 시간
event	데이터의 유형 ex)점수등록, 상점이용
properties	Event에 대한 세부 내용
action	해당 플레이를 완료 했는지 여부
build	플레이한 차량의 세부 속성
properties.id	플레이한 스테이지 속성
properties.package	플레이 도중 획득한 보너스
properties.reward	최종 플레이 결과에 대한 보상(점수)
propertie.balance	상점 구매 전 보유한 게임머니
propertie.price	상품 구매 가격
propertie.remaining	상점 구매 후 남은 게임머니
propertie.type	상품 구매 내역
uid	유저의 고유한 ID

표 6 주요 데이터의 속성 설명

위와 같이 JSON으로 저장되어 있는 데이터를 전처리 과정을 통해 Game 1과 같이 최종적으로 완성된 전처리 데이터 테이블로 만들기 위해서는 원본 데이터 파일에서 몇 개의 속성만 검출해서 사용해야 한다. 총 3개의 게임에 대한 정형화된 유저 이탈 예측 연구를 하기 위해서는 Game 1, Game 2, Game 3의 전처리된 데이터 테이블의 속성이 비슷해야한다. Game 2에서는 두개의 전처리된 데이터 테이블을 만들기로 한다. [표 7]은 게임 플레이 관련 전처리된 테이블을 나타내고 있으며, [표 8]은 상점이용 관련 전처리된 테이블을 나타내고 있다.

id	device	score	time
56124	63BFD298E47F9010129DAB28C9390E	4363	1406267278
56125	63BFD298E47F9010129DAB28C9390E	334	1406267280
56126	E47F9010645FA67BBD8C938C938C93	221	1406267281
56127	E47F9010645FA67BBD8C938C938C93	446	1406267290

표 7 게임 플레이 관련 전처리된 테이블 예시

게임 플레이 시 저장되는 데이터 테이블의 propertie.reward 속성은 하나의 게임이 끝나고 최종으로 받게되는 보상으로써 해당 플레이에 대한 점수와 같은 역할을 한다. 이는 Game 1의 score와 같은 개념이며, 위와 같이 게임 플레이 관련 전처리된 데이터 테이블의 score 속성으로 추출 가능하다.

id	device	price	time
74162	A7E0855A49F0F28E955748035008D6	500	1406263428
74163	A7E0855A49F0F28E955748035008D6	800	1406263430
74164	CE26DEF6B3305E9554748E55E20463	3000	1406263452
74165	CE26DEF6B3305E9554748E55E20463	5500	1406263480

표 8 상점 관련 전처리된 테이블 예시

상점 관련 전처리된 테이블에서 price는 상점 이용시 저장되는 데이터 테이블의 propertie.price 속성을 가지고 추출하였다. 정형화된 유저 이탈 예측을 위해 공통된 Feature를 추출하기 위해서는 Game 1에서 전처리된 테이블로 만들었던 device, score, time 속성이 들어간 데이터 테이블로 만들어야 하고 Game 2에서는 게임 플레이 관련

전처리된 테이블이 이를 서포트 해주고 있다. 그러나 상점 관련 전처리된 테이블은 Game 1에는 존재하지 않는 테이블이다. 이는 유저 이탈 예측 모델 설계시 Game 2에 종속적으로 적용되는 Feature를 추출하기 위해서 만들었다.

Game 3

Game 3에서 직접 제공하는 Api를 통해서 가져온 데이터는 JSON 포멧으로 제공되고 있다. 또한, 데이터 테이블은 2가지로 나뉘어서 제공되고 있다. [그림 23]은 게임 플레이에 관한 데이터를 나타내고 있고, [그림 24]은 유저의 개인 기록에 관한 데이터를 나타내고 있다. 또한, [표 9]는 주요 데이터 속성에 대해 설명하고 있다. 데이터가 2개로 나뉘어져 있기 때문에 데이터 전처리를 하기 위해서는 서로 다른 2개의 데이터의 연결고리를 찾아 통합을 시켜야한다.


```
"1":{
        "server":"tagpro-chord.koalabeast.com",
        "port":8008,
        "official":true,
        "group":"",
        "date":1432576197,
        "timeLimit":12,
        "duration":22562,
        "mapId":6,
        "teams":[
            {
                "name":"Red",
                "score":1,
                …(truncated)
            },
            {
                "name":"Blue",
                "score":3,
                …(truncated)
            }
        ]
    },
```


그림 23 Game 3 게임 플레이 데이터 구조

```
"1":[
        {
            "auth":true,
            "name":"RoDyMaRy",
            "flair":105,
            "degree":99,
            "score":31,
            "points":25,
            "team":1,

        },
        {
            "auth":true,
            "name":"BartimaeusJr",
            "flair":86,
            "degree":0,
            "score":56,
            "points":0,
            "team":2,
        },
…(truncated)
```


그림 24 Game 3 유저 개인 기록 데이터 구조

Attribute	내용
timeLimit	한 판의 플레 당 설정된 시간제한
duration	게임 종료까지 플레이 된 시간
teams.name	팀의 색
teams.score	팀이 획득한 점수
name	개인의 이름
score	해당 게임 플레이에서 획득한 점수
points	팀이 승리한 경우 포인트 획득
team	소속된 팀

표 9 주요 데이터의 속성 설명

앞서 설명한바와 같이 Game 3 데이터는 2개로 나뉘어져 있다. 두 데이터 구조 간의 연결고리를 가지고 통합을 해야 하는데, 양 쪽 구조 모두 데이터 상위 계층에 해당 플레이 id가 적혀져 있다. 해당 플레이 id가 서로간의 연결고리가 되어 2개의 데이터를 통합 하도록 한다.

Game 2에서 설명한 대로 최종적으로 전처리가 완료된 데이터 테이블을 만들기 위해서는 기존에 Game 1, Game 2에서 사용된 테이블의 속성(device, score, time)에 맞게 테이블을 만들어야 한다. Game 3에서 score로 추출 가능한 속성은 유저 개인 기록 데이터 테이블의 score로써 게임 플레이 동안 획득 한 점수를 뜻한다. [표 10]은 최종적으로 만든 전처리된 데이터 테이블의 예시이다.

id	device	score	point	duration	time
24882	ksw29zz	500	15	24685	1406651521
24883	ksw29zz	800	0	15221	1406663728
24884	sunnyjun	3000	25	31186	1406683515
24885	sunnyjun	5500	0	42215	1406691521

표 10 Game 3 전처리된 테이블 예시

device 속성은 유저의 실제 닉네임으로 들어가 있다. 기존의 전처리된 데이터 테이블의 포멧을 유지하기 위해 속성명은 바꾸지 않았다. 또한, Game 2와 마찬가지로 Game 3에 종속적인 Feature를 추출하기 위해서 기존 전처리된 데이터 테이블에 point행과 duration행을 추가하도록 한다. point는 팀이 승리한 경우 획득하는 수치로 패배할 경우에는 수치가 0인채로 획득이 불가하다. 이를 통해 Game 3가 가지고 있는 팀에 대한 승패에 대한 Game 3에만 종속적인 Feature를 만들 수가 있다. 그리고 duration값을 가지고 게임 자체에 대한 플레이 시간을 저장한 값으로 실제 게임을 얼마나 해서 승리 혹은 패배를 했는지에 대한 Game 3에 종속적인 Feature를 만들 수 있다.

제 3 절 Feature 정의

유저 이탈 예측 모델을 설계 함에 있어서 Feature 설정은 필수적이며 Feature들의 성능에 따라 예측 성능도 바뀌게 된다. 이에 어떤 Feature를 정의하고 선택하느냐는 아주 중요한 연구주제가 되고 있다[31].

본 연구에서는 3개의 게임에 대한 공통된 10개의 Feature들과 Game 2, Game 3에 특정적으로 적용되는 4개의 전용 Feature들을 정의한다. 3개의 게임에 공통된 10개의 Feature는 [표 11]와 같다.

Feature 명	설명	데이터형태
playCount	유저의 게임 플레이 횟수	Integer
bestScore	가장 높은 점수	Integer
consecutivePlayRatio	전체 플레이 대비 연속된 플레이(15분) 비율	Float
worstScore	가장 나쁜 점수	Integer
meanScore	평균 점수	Integer
sdScore	점수들의 표준편차	Float
bestSubMeanRatio	(최고점수 – 평균점수) / 평균 점수	Float
bestSubMeanCount	(최고점수 – 평균점수) / 플레이 수	Float
bestScoreIndex	(최고점수 때의 플레이 번째) / 플레이 수	Integer
activeDuration	마지막 플레이 시간 – 첫 플레이 시간	Integer

표 11 총 3개의 게임에 대한 공통된 Feature

위에서 설명한 3개의 게임에서 공통으로 적용되는 Feature들은 대부분 게임 스코어와 시간과 연관된 Feature들로써 전처리된 데이터 테이블의 속성인 device, score, time 속성들로 추출이 가능하다. 또한, [표 12]는 각 게임 별 전용 Feature들에 대한 설명이다.

소속	Feature 명	설명	데이터형태
Game 2	purchaseCount	상점 구매 횟수	Integer
	bestPurchase	상점에서 가장 많이 지불한 금액	Integer
Game 3	winRatio	승리율	Float
	gameDurationMean	평균 게임 시간	Float

표 12 각 게임 별 전용 Feature

Game 2의 나머지 전처리된 테이블인 상점 관련 전처리 테이블과 Game 3의 마지막 행인 points 속성을 가지고 각 게임에 종속된 Feature를 정의하였다. 본 3절에서는 위에서 소개된 총 14개의 Feature들에 대한 구체적인 정의와 기본 통계를 통해서 각 Feature들에 대해 심도있게 살펴보도록 한다.

1. 공통 Feature

playCount

플레이횟수는 유저가 해당 관찰 기간 동안 게임 플레이를 한 횟수이다. 유저가 해당 관찰 기간 동안 몇 번 플레이하였는지에 대한 Feature이다. playCount를 추출하기 위해서는 유저 별로 관찰 기간 범위 안에 데이터를 나열한 후 데이터 열의 개수를 카운팅한다. 공식은 다음과 같다[공식 1].

$$playCount = n$$

공식 1 Feature 산출 공식: playCount

또한, [그림 25]은 playCount를 추출하는 것에 대한 예시 설명이다.

User A

playCount = 9

score	time
4	2015-10-29 23:56:55
252	2015-10-30 00:08:11
169	2015-10-30 00:10:08
238	2015-10-30 00:15:01
184	2015-10-30 00:15:48
208	2015-10-30 00:19:38
353	2015-10-30 00:33:23
586	2015-10-30 07:48:13
472	2015-10-30 09:45:21

User B

playCount = 10

score	time
283	2015-05-24 16:00:11
469	2015-05-24 16:01:22
234	2015-05-24 16:13:34
338	2015-05-25 09:31:43
361	2015-05-27 14:44:19
297	2015-05-27 22:14:58
306	2015-05-27 22:16:31
243	2015-05-28 18:33:32
474	2015-06-03 13:32:17
413	2015-06-03 15:42:55

그림 25 Feature 정의: playCount

그림에서 보는 바와 같이, User A의 데이터의 전체 열의 수가 9번이므로 플레이횟수는 9번이고, User B는 데이터의 전체 열의 수가 10번이므로 플레이횟수는 10번으로 정한다. 단, 그림에 나와있는 User A와 User B의 시간값은 관찰 기간 안에 속해 있어야 한다.

아래 그래프는 위에서 정의한 playCount를 가지고 게임 별로 전체 유저를 대상으로 관찰 기간 별 평균 playCount의 변화에 대한 그래프를 그려보았다[그림 26].

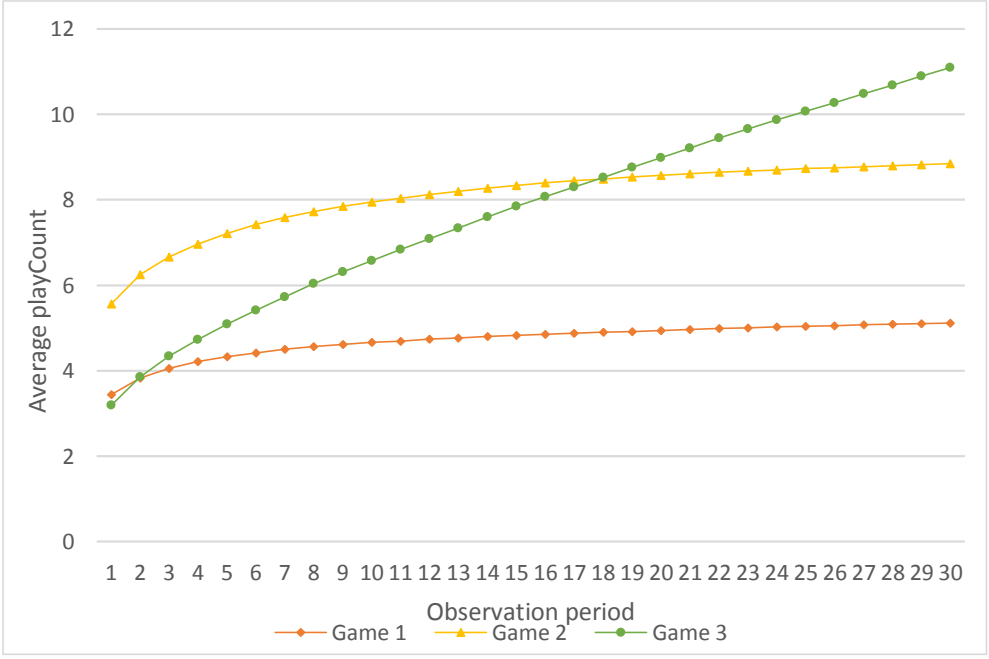

그림 26 관찰 기간 별 playCount의 변화

Game 1은 관찰 기간이 길어짐에 따라서 평균 플레이 횟수의 증가량이 점점 줄어든다. 이는 Game 1의 유저들이 시간이 갈수록 점점 플레이하는 횟수가 급격히 줄어든다고 볼 수 있다. Game 2는 평균 플레이 횟수가 Game 1과 Game 3보다 높고 Game 1에 비해서 처음 10일 동안은 플레이 하는 유저들이 많이 존재를 하였다. 그러나 10일 이후부터 Game 1과 비슷하게 플레이 하는 유저들이 급격히 줄어들었다. Game 3의 경우는 관찰 기간 첫날 세개의 게임 중에 평균 플레이 횟수가 가장 적었지만 관찰 기간이 길어질수록 평균 플레이 횟수가 꾸준히 증가되었다. 이는 Game 3이 게임 플레이 시간이 다른 두게임에 비해서 길어서 첫 하루 동안엔 플레이 횟수가 적지만 게임성이 다른 두 게임에 비해 좋기 때문에 위와 같은 그래프가 나왔다고 해석될 수 있다. 또한, 1절에서 설명한 Game 3의 이탈률 그래프에서도 Game 1, Game 2의 이탈률이 30일 이후엔 약 95%인데 반해 Game 3는 약 80%가 나오는데, 이는 위와 같은 해석을 할 수 있는 근거가 될 수 있다[그림 16].

bestScore

bestScore는 해당 관찰 기간 동안 최고 점수이다. 관찰 기간 안에 있는 점수 데이터 중 가장 높은 데이터를 말하는다. bestScore의 공식은 다음과 같다[공식 2].

$$bestScore = Max(score)$$

공식 2 Feature 산출 공식: bestScore

[그림 27]에서 bestScore를 유저 데이터에서 추출하는 것을 도식화하여 나타내었다.

User A

score	time
4	2015-10-29 23:56:55
252	2015-10-30 00:08:11
169	2015-10-30 00:10:08
238	2015-10-30 00:15:01
184	2015-10-30 00:15:48
208	2015-10-30 00:19:38
353	2015-10-30 00:33:23
586	2015-10-30 07:48:13
472	2015-10-30 09:45:21

bestScore = 586

User B

score	time
283	2015-05-24 16:00:11
469	2015-05-24 16:01:22
234	2015-05-24 16:13:34
338	2015-05-25 09:31:43
361	2015-05-27 14:44:19
297	2015-05-27 22:14:58
306	2015-05-27 22:16:31
243	2015-05-28 18:33:32
474	2015-06-03 13:32:17
413	2015-06-03 15:42:55

bestScore = 474

그림 27 Feature 정의: bestScore

위 그림을 보면 User A의 속성 중 Score에서 가장 높은 점수는 586점이다. 또한, User B의 가장 높은 점수는 474점이다. 아래 그래프에서는 위에서 정의한 bestScore를 가지고 게임 별 전체 유저들을 대상으로 관찰 기간 별 평균 bestScore의 변화에 대한 그래프를 그려보았다[그림 28, 그림 29, 그림 30].

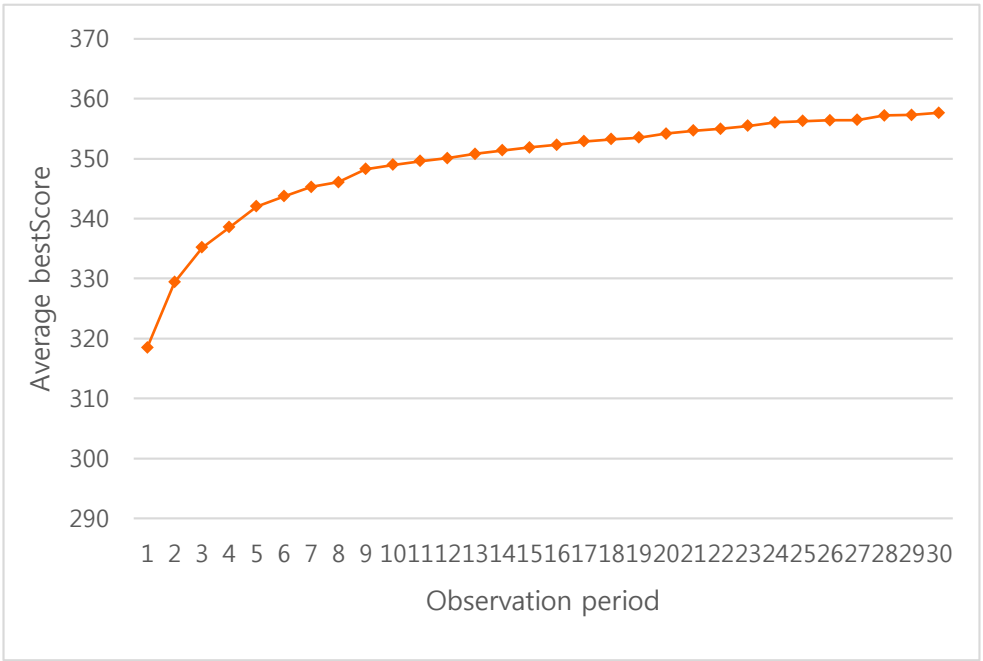

그림 28 Game 1 관찰 기간 별 bestScore의 변화

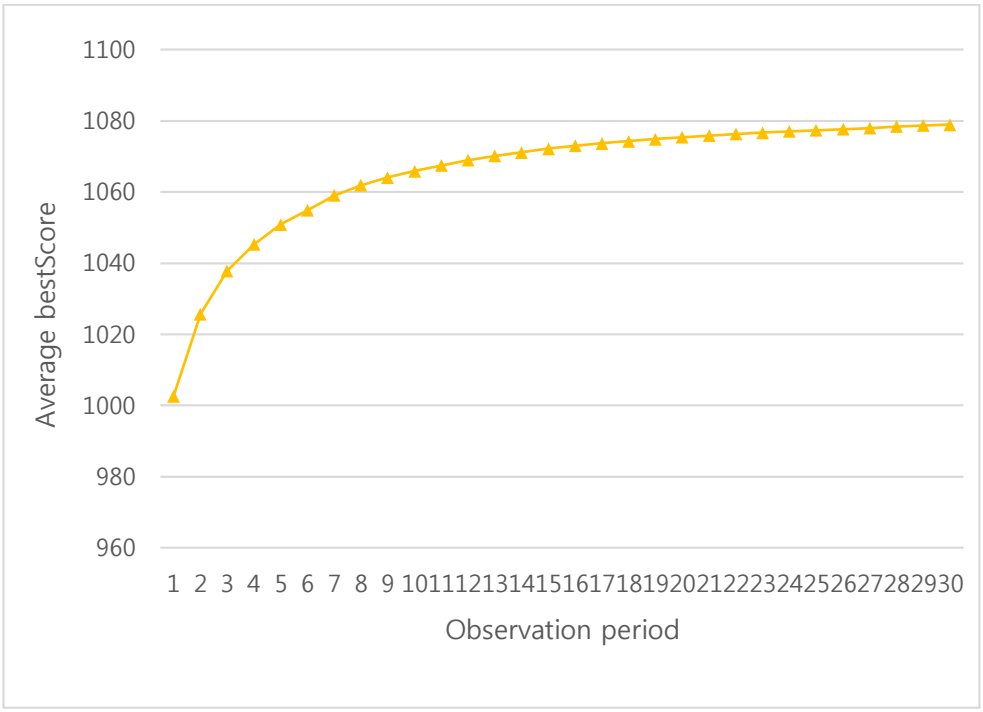

그림 29 Game 2 관찰 기간 별 bestScore의 변화

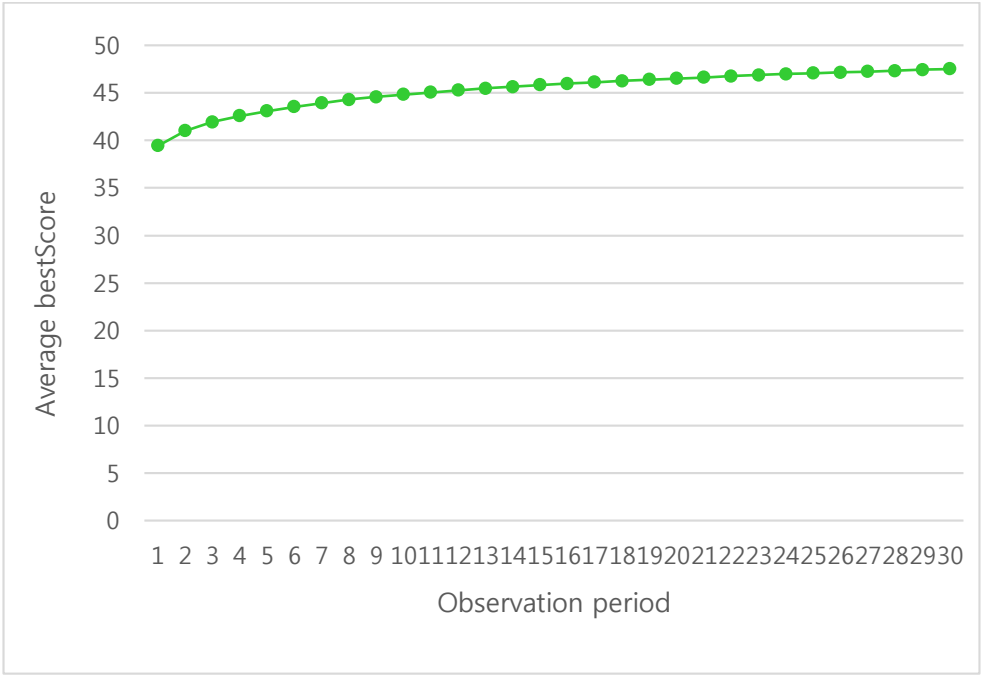

그림 30 Game 3 관찰 기간 별 bestScore의 변화

Game 1, Game 2 그래프에서 처음 7일 동안의 관찰 기간에서는

평균 최고 점수 증가 폭이 높아지고 일주일이 넘어가면서 평균 최고 점수가 수렴하게된다. Game 3는 Game 1, Game 2에 비해서 평균 최고 점수 증가 폭이 작다. 이는 Game 1, Game 2은 유저들이 게임 초반에 학습을 통해서 최고 점수를 기록을 일정 기간이 지나면 더 이상 최고 점수를 기록하지 못하게 된다고 해석이 가능하다. 또한, Game 3는 유저들의 최고 점수에 대한 증가 폭이 비교적 적은데 이는 게임 설계상 최고 점수보다는 게임의 승패에 더욱 초점을 두고 있고 유저도 최고 점수를 올리는 플레이를 하지 않는다는 해석을 할 수 있다.

consecutivePlayRatio

consecutivePlayRatio은 해당 관찰 기간 동안의 전체 플레이 대비 연속된 플레이 비율이다. 유저가 얼마나 몰아서 하는지를 보기위한 Feature이다. 관찰기간안의 플레이 기록이 있고 하나의 플레이 기록과 이후 플레이 기록과의 시간의 차가 15분이하이면 연속된 플레이가 된다. 연속된 플레이의 개수를 구하고, 전체 플레이 대비 연속된 플레이의 비율을 연속성 값으로 정한다. Game 1, Game 2의 경우 한번의 게임플레이는 고득점 유저인 경우 3~5분정도 걸리고, Game 3의 경우 평균 플레이 시간이 5분이고 최대 12분의 플레이 시간의 제한이있다. 이런 이유로 연속된 플레이를 15분 이하로 잡아두었다. [그림 31]에서 유저 데이터에서 consecutivePlayRatio 값을 추출하는 것을 보여준다.

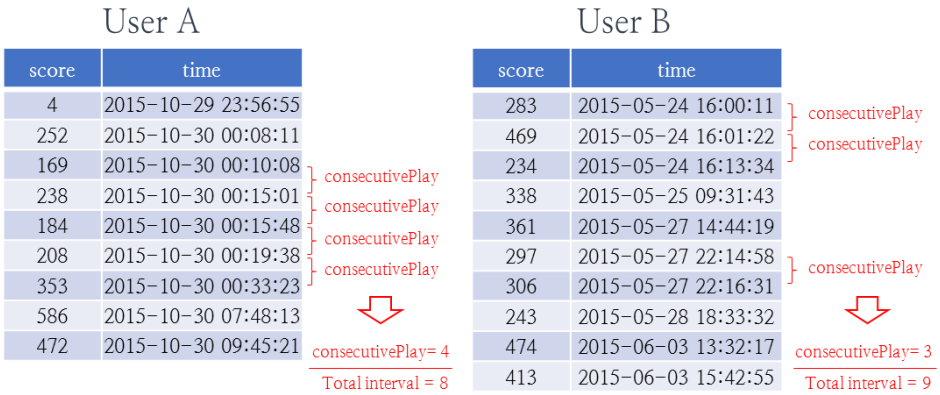

그림 31 Feature 정의: consecutivePlayRatio

위 그림에 설명되어 있듯이 User A는 전체 플레이 간격 8개 중에서 연속된 플레이가 4번이 된다. 그래서 User A의 consecutivePlayRatio는 $\frac{4}{8}$인 0.5가된다. User B는 전체 플레이 간격 9개 중에 연속된 플레이가 3번이 되기 때문에 consecutivePlayRatio는 $\frac{3}{9}$ 인 0.333이 된다. 아래 그래프에서는 위에서 정의한 consecutivePlayRatio를 가지고 게임 별 전체 유저들을 대상으로 관찰 기간 별 평균 consecutivePlayRatio의 변화에 대한 그래프를 그려보았다[그림 32].

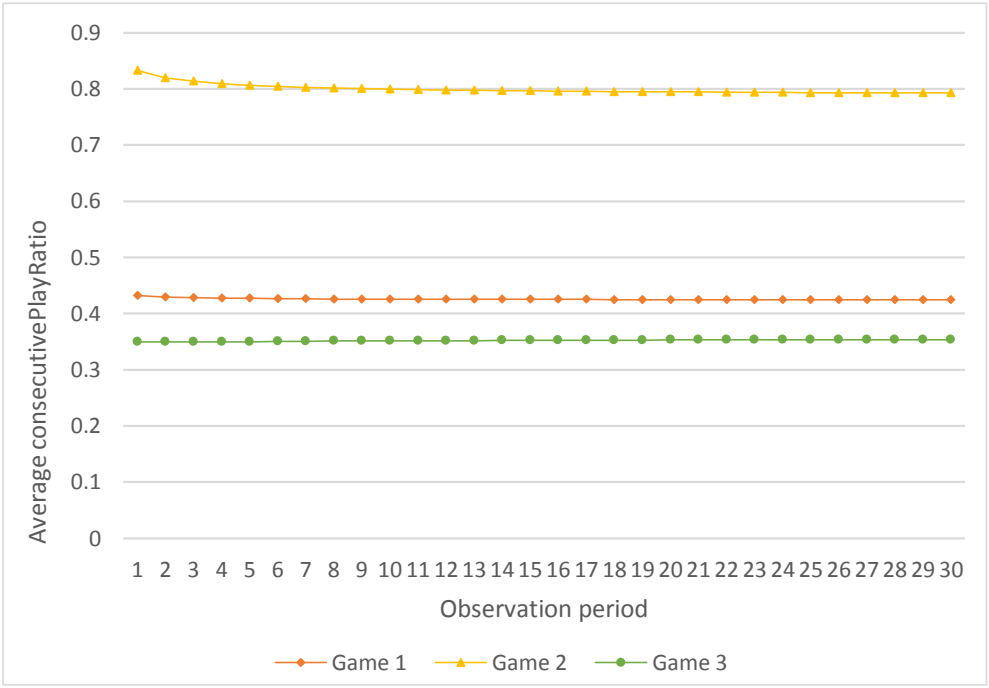

그림 32 관찰 기간 별 consecutivePlayRatio의 변화

Game 1은 평균 consecutivePlayRatio가 약 0.43정도로 유저들의 플레이 중 약 43%가 몰아서 하는 플레이라고 해석이 가능하다. 또한, Game 3는 평균 consecutivePlayRatio가 약 0.35정도로 유저들의 플레이 중 약 35%가 몰아서 하는 플레이라고 해석이 가능하다. Game 2의 경우엔 consecutivePlayRatio가 약 0.8 정도로 유저들의 플레이 중 약 80%가 몰아서 하는 플레이라고 볼 수 있다. 이런 결과는 앞서 설명하였던 이탈률과 연결해서 해석해 볼 수

있다[그림 16]. Game 2과 Game 3의 첫날 이탈률은 약 60%로 비슷한 이탈율로 시작하였으나 30일이 지난 후의 이탈률은 Game 2는 약 95%인데 비해 Game 3는 약 80%로 많은 이탈률의 차이를 보여줬다. 이는 Game 2의 유저들이 Game 3에 비해 몰아서 플레이를 하는 경향이 심했으며, 몰아서 하는 유저들은 쉽게 게임에 지겨워하여 이탈을 했다는 해석이 가능하다.

worstScore

worstScore는 해당 관찰 기간 동안 최저 점수이다. 유저 별로 관찰기간 안에 데이터를 나열에 놓고 그 중 가장 낮은 점수 수치를 가지고 worstScore로 추출한다. worstScore의 공식은 다음과 같다[공식 3].

$$worstScore = Min(score)$$

공식 3 Feature 산출 공식: worstScore

[그림 33]에서 worstScore를 전처리된 데이터 테이블에서 추출하는 것을 도식화하여 나타내었다.

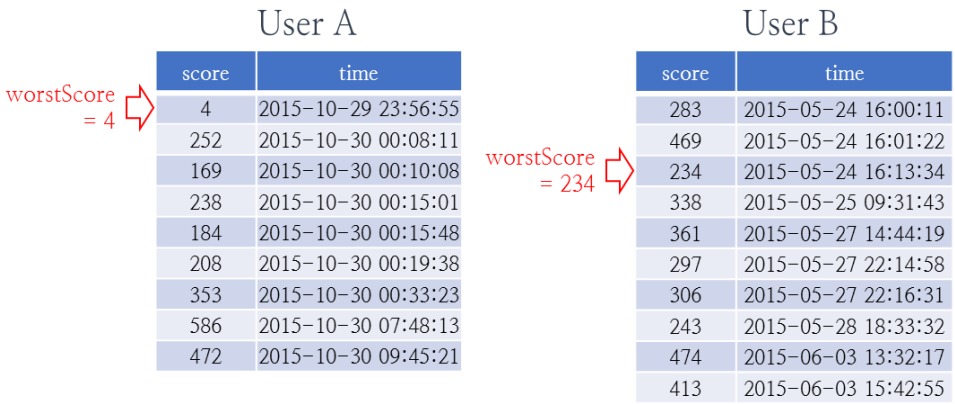

그림 33 Feature 정의: worstScore

위 그림을 보면 User A의 가장 낮은 점수는 4점으로써 이 점수가 User A의 worstScore이다. User B도 마찬가지 방식으로 worstScore가 234이다. 아래 그래프에서는 위에서 정의한 worstScore를 가지고 게임 별 전체 유저들을 대상으로 관찰 기간 별 평균 worstScore의 변화에 대한 그래프를 그려보았다[그림 34].

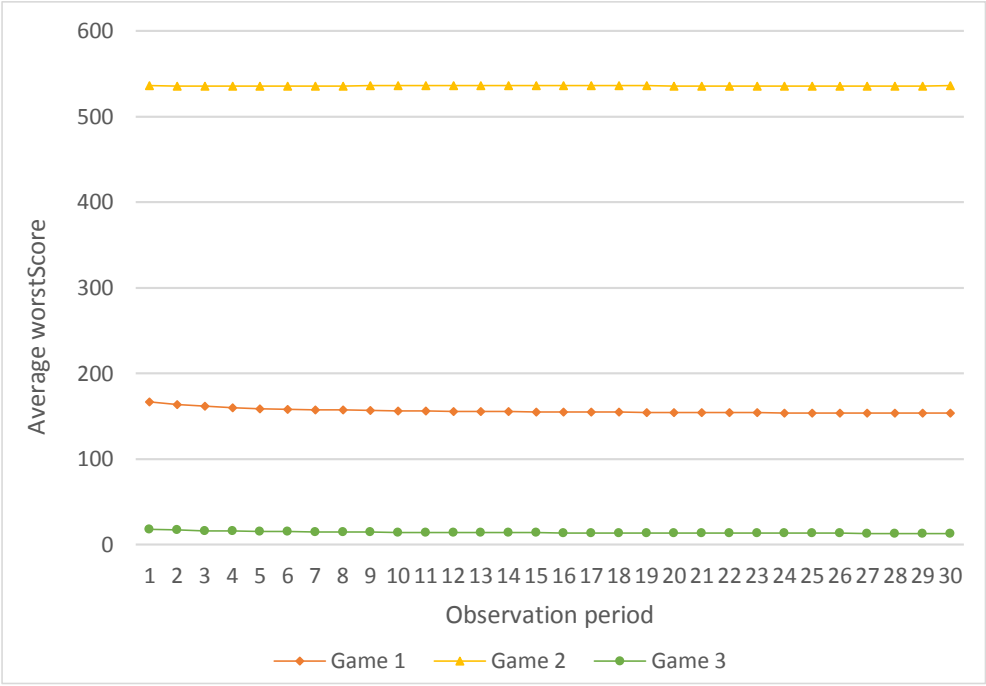

그림 34 관찰 기간 별 worstScore의 변화

Game 1, Game 2, Game 3의 평균 worstScore의 변화는 아주 적다. 그러나 여기서 게임상의 특이점이 하나 있는데, Game 3에서는 0 미만의 점수가 존재한다는 점이다. 팀에 도움이 되지 않는 행동들을 하면 0 미만의 마이너스 점수를 기록하게 된다[32]. 이에 마이너스 점수를 기록 한 유저들만 살펴볼 필요가 있는데, 아래 2개의 그래프는 전체 인원 대비 마이너스 점수를 기록 한 유저의 비율의 변화를 나타낸 그래프와 마이너스 점수 기록 유저의 평균 worstScore 점수 그래프이다[그림 35, 그림 36].

그래프에서 보는 바와 같이 관찰 기간이 지남에 따라 마이너스 점수의 유저의 인구 비율이 점점 늘어난다. 또한, 평균

마이너스 점수도 점차 낮아진다. 마이너스 플레이를 하는 유저는 일종의 변칙 플레이유저로 볼 수 있다. 이러한 유저들이 게임의 결과를 좀더 변칙적으로 만들 수 있고, 재미난 게임 플레이를 만들어 줄 수 있다. 이에 마이너스 점수 유저에 대한 그래프는 앞서 설명하였던 Game 3의 낮은 이탈률에 대한 근거가 되는 그래프가 될 수 있다는 해석이 가능하다.

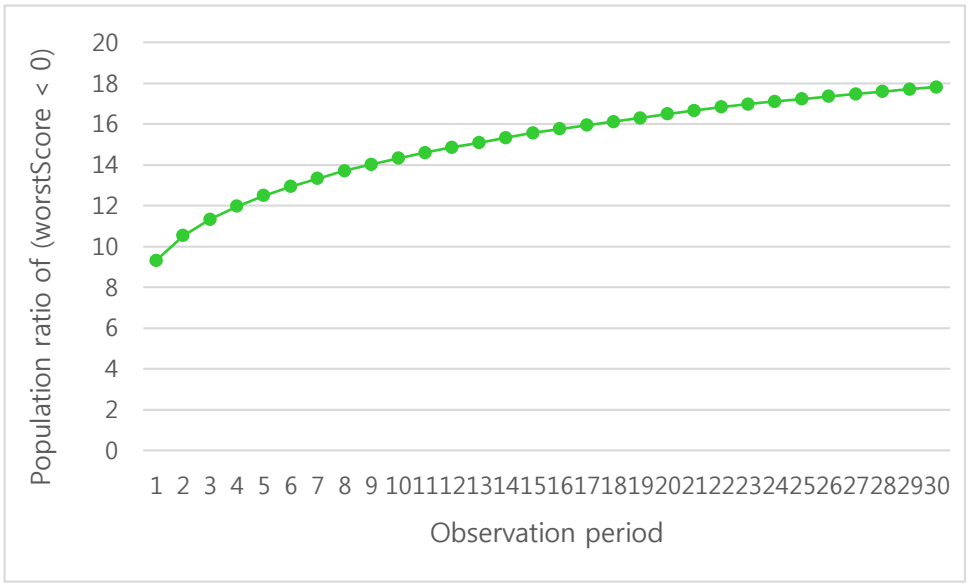

그림 35 관찰 기간 별 전체 유저 대비 마이너스 점수 유저 인구 비율

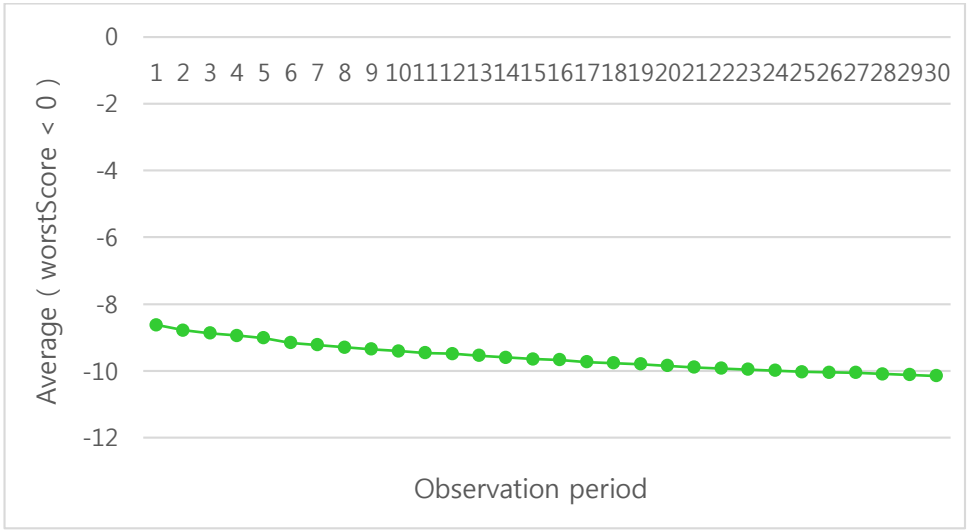

그림 36 관찰 기간 별 마이너스 점수 유저의 평균 점수

meanScore

meanScore는 해당 관찰 기간 동안의 점수들의 평균 점수이다. 유저 별로 게임 내에서 기록한 점수들을 나열하고 관찰 기간내에 있는 점수들을 가지고 평균을 내어 meanScore를 추출한다. 추출하는데 쓰이는 meanScore의 공식은 아래와 같다[공식 4].

$$meanScore = \frac{\sum_{i=1}^{n} score_i}{n}$$

공식 4 Feature 산출 공식: meanScore

[그림 37]에서 전처리된 데이터를 가지고 meanScore를 추출하는 방식에 대해서 도식화하여 나타내었다.

User A

score	time
4	2015-10-29 23:56:55
252	2015-10-30 00:08:11
169	2015-10-30 00:10:08
238	2015-10-30 00:15:01
184	2015-10-30 00:15:48
208	2015-10-30 00:19:38
353	2015-10-30 00:33:23
586	2015-10-30 07:48:13
472	2015-10-30 09:45:21

$\frac{\sum_{i=1}^{n} score_i}{n} = \frac{2466}{9} = 274$

User B

score	time
283	2015-05-24 16:00:11
469	2015-05-24 16:01:22
234	2015-05-24 16:13:34
338	2015-05-25 09:31:43
361	2015-05-27 14:44:19
297	2015-05-27 22:14:58
306	2015-05-27 22:16:31
243	2015-05-28 18:33:32
474	2015-06-03 13:32:17
413	2015-06-03 15:42:55

$\frac{\sum_{i=1}^{n} score_i}{n} = \frac{3418}{10} = 341.8$

그림 37 Feature 정의: meanScore

위에서 정의한 meanScore를 가지고 게임 별 전체 유저들을 대상으로 관찰 기간 별 평균 meanScore의 변화에 대한 그래프를 [그림 38]와 같이 그려보았다.

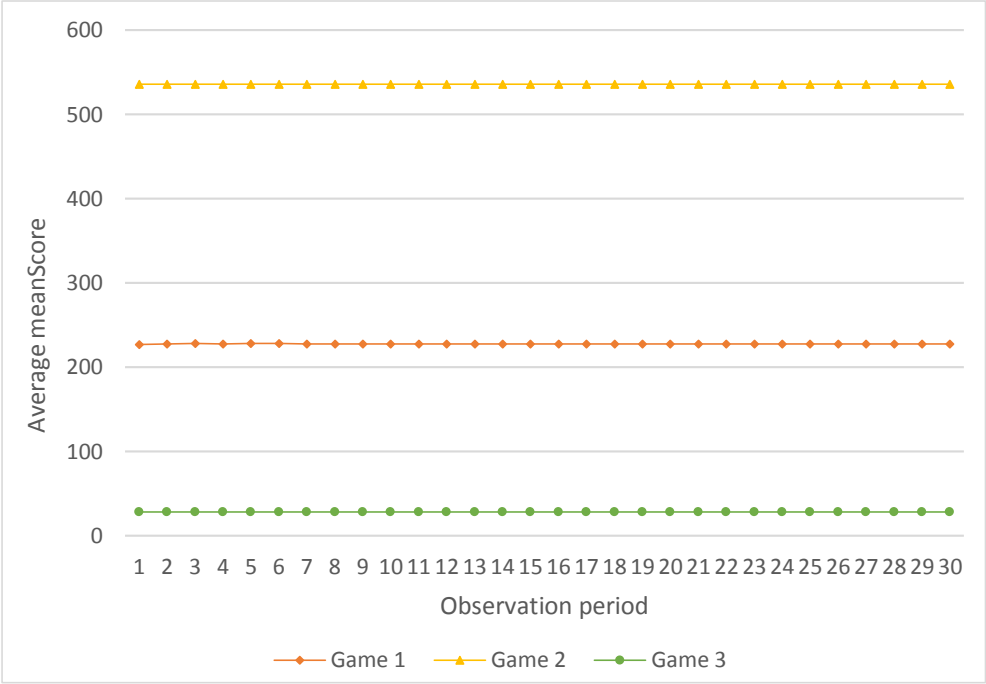

그림 38 관찰 기간 별 meanScore의 변화

위의 그래프를 살펴보면 세개의 게임 모두 다 그래프의 변화가 없다. 만약 관찰 기간이 지남에 따라서 평균 점수의 변화가 크게 보인다면, 게임자체의 점수 시스템 설계나 기획이 바뀌어서 유저들의 전체적인 점수 획득의 변화가 온 경우라고 해석이 가능하다. 그러기에 위 그래프는 관찰 기간 동안에 특별한 게임의 점수 시스템 설계나 기획의 변화가 없다는 것을 보여준다.

sdScore

sdScore는 유저들의 점수의 표준편차이다. 한 유저의 점수 기록이 꾸준히 일정 점수를 유지하는지, 혹은 매 플레이마다 기록하는 점수가 들쑥날쑥 한 지를 보기 위한 Feature이다. sdScore가 높다면 유저의 점수의 편차가 높아 들쑥날쑥하다는 것을 의미를 하고 sdScore가 낮다면 편차가 낮아 점수가 평이하다는 것을 의미한다. 유저별로 관찰 기간 안에 데이터를 나열하고 모든 점수에대한 표준편차를 sdScore로 추출한다. 추출하는것에 대한 sdScore의 공식은 아래와 같다[공식 5].

$$sdScore = \sqrt{\frac{\sum_{i=1}^{n}(score_i - \overline{score})^2}{n-1}}$$

공식 5 Feature 산출 공식: sdScore

[그림 39]에서 전처리된 테이블에서 sdScore를 추출하는 것에 대해 도식화하여 나타내었다.

User A

score	time
4	2015-10-29 23:56:55
252	2015-10-30 00:08:11
169	2015-10-30 00:10:08
238	2015-10-30 00:15:01
184	2015-10-30 00:15:48
208	2015-10-30 00:19:38
353	2015-10-30 00:33:23
586	2015-10-30 07:48:13
472	2015-10-30 09:45:21

$\sqrt{\frac{\sum_{i=1}^{n}(score_i - \overline{score})^2}{n-1}} \fallingdotseq 173.5475$

User B

score	time
283	2015-05-24 16:00:11
469	2015-05-24 16:01:22
234	2015-05-24 16:13:34
338	2015-05-25 09:31:43
361	2015-05-27 14:44:19
297	2015-05-27 22:14:58
306	2015-05-27 22:16:31
243	2015-05-28 18:33:32
474	2015-06-03 13:32:17
413	2015-06-03 15:42:55

$\sqrt{\frac{\sum_{i=1}^{n}(score_i - \overline{score})^2}{n-1}} \fallingdotseq 86.43404679$

그림 39 Feature 정의: sdScore

위에서 정의한 sdScore를 기반으로 게임 별 전체 유저들을 대상으로 관찰 기간 별 평균 sdScore의 변화에 대한 그래프를 [그림 40, 그림 41, 그림 42]와 같이 그려보았다.

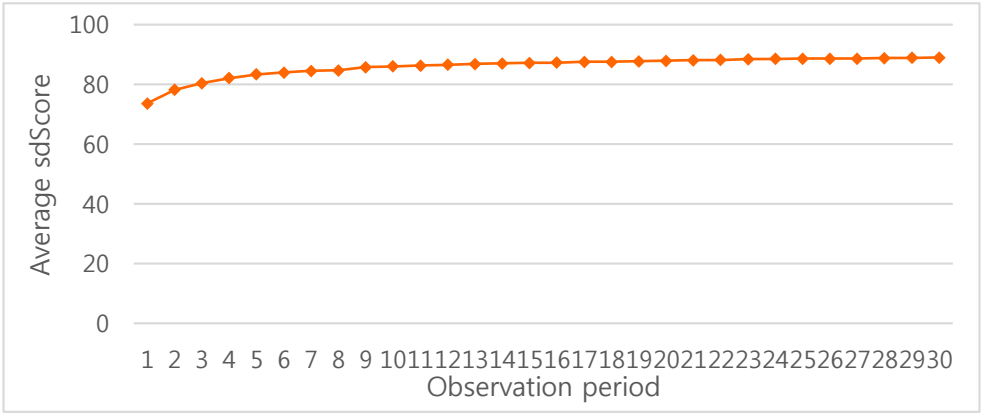

그림 40 Game 1 관찰 기간 별 sdScore의 변화

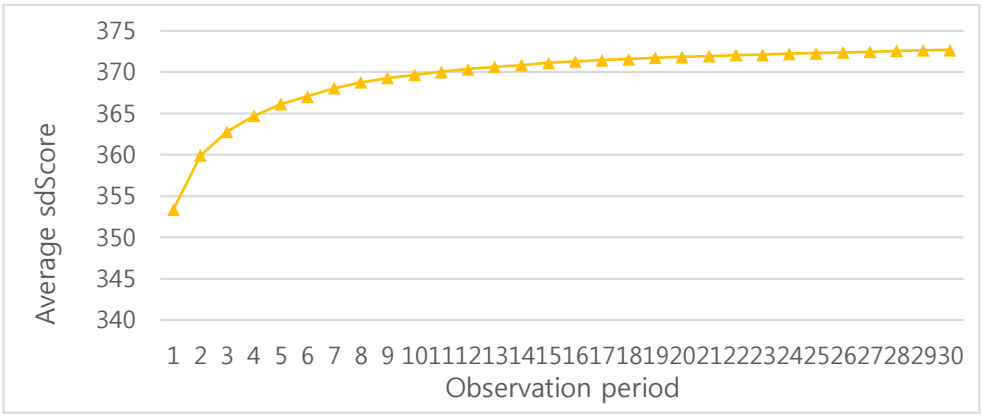

그림 41 Game 2 관찰 기간 별 sdScore의 변화

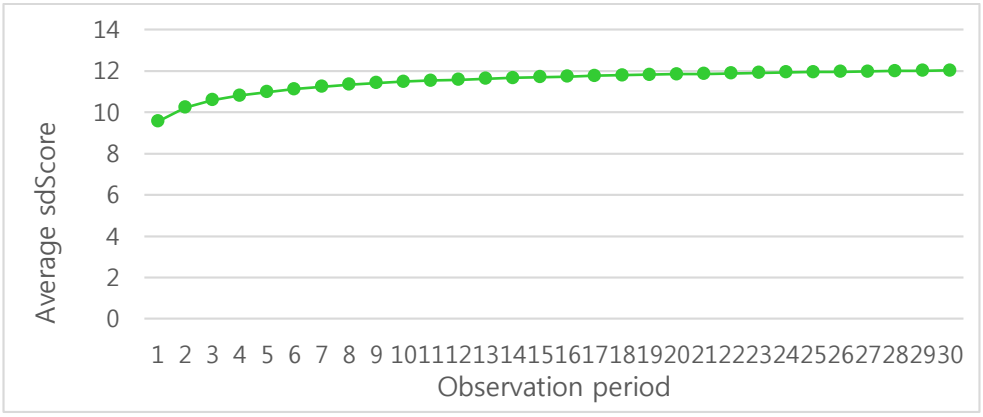

그림 42 Game 3 관찰 기간 별 sdScore의 변화

위의 그래프들을 살펴보면, 세 게임들대한 유저들의 평균 표준편차의 증가가 관찰 기간 처음 일주일 동안은 확실히 들어났지만 관찰 기간이 길어짐에 따라 수렴하는 모양을 띄고 있다. 특히 평균 표준편차의 증가량를 살펴보면 1일부터 7일까지 증가한 양과 8일부터 30일동안 증가한 양의 비율이 Game 2가 Game 1, Game 3보다 월등히 많았다. 이는 Game 2의 유저들의 첫 일주일동안 점수 기록이 Game 1, Game 3보다 더 들쑥날쑥 했다는 것을 보여주고 있다. 이는 앞서 설명한 관찰기간 별 이탈율에서 Game 2가 첫 일주일동안 급격한 이탈을 보여주는 것을 설명해준다[그림 19]. 유저의 점수 기록이 들쑥날쑥한 정도가 크면 유저들이 게임의 흐름을 이해하지 못하여 이탈을 할 수 있다는 해석이 가능하다.

bestSubMeanRatio

bestSubMeanRatio는 최고 점수의 득점이 평소 실력에 비해 얼마나 차이가 나는지를 알기 위한 Feature이다. 유저 별로 게임 내에서 기록한 점수들 중 최고 점수에 평균 점수를 뺀 값에 평균 점수를 나눈 값을 bestSubMeanRatio로 추출한다. bestSubMeanRatio의 공식은 아래와 같다[공식 6].

$$bestSubMeanRatio = \frac{Max(score) - \overline{score}}{\overline{score}}$$

공식 6 Feature 산출 공식: bestSubMeanRatio

위 공식에서와 같이 bestSubMeanRatio값이 크면 유저의 최고 점수가 평소의 실력 대비 많이 나오게된 것이고, 값이 낮으면 유저의 최고 점수는 평소의 실력과 별 차이가 없다는 걸 알 수 있다.

[그림 43]에서 전처리된 테이블에서 bestSubMeanRatio를 추출하는 것에 대해 도식화하여 나타내었다.

User A

score	time
4	2015-10-29 23:56:55
252	2015-10-30 00:08:11
169	2015-10-30 00:10:08
238	2015-10-30 00:15:01
184	2015-10-30 00:15:48
208	2015-10-30 00:19:38
353	2015-10-30 00:33:23
586	2015-10-30 07:48:13
472	2015-10-30 09:45:21

$\frac{Max(score) - \overline{score}}{\overline{score}} = \frac{586 - 274}{274} \fallingdotseq 1.1387$

User B

score	time
283	2015-05-24 16:00:11
469	2015-05-24 16:01:22
234	2015-05-24 16:13:34
338	2015-05-25 09:31:43
361	2015-05-27 14:44:19
297	2015-05-27 22:14:58
306	2015-05-27 22:16:31
243	2015-05-28 18:33:32
474	2015-06-03 13:32:17
413	2015-06-03 15:42:55

$\frac{Max(score) - \overline{score}}{\overline{score}} = \frac{474 - 341.8}{341.8} \fallingdotseq 0.3463$

그림 43 Feature 정의: bestSubMeanRatio

위에서 정의한 bestSubMeanRatio 를 기반으로 게임 별 전체 유저들을 대상으로 관찰 기간 별 평균 bestSubMeanRatio 의 변화에

대한 그래프를 [그림 44]와 같이 그려보았다.

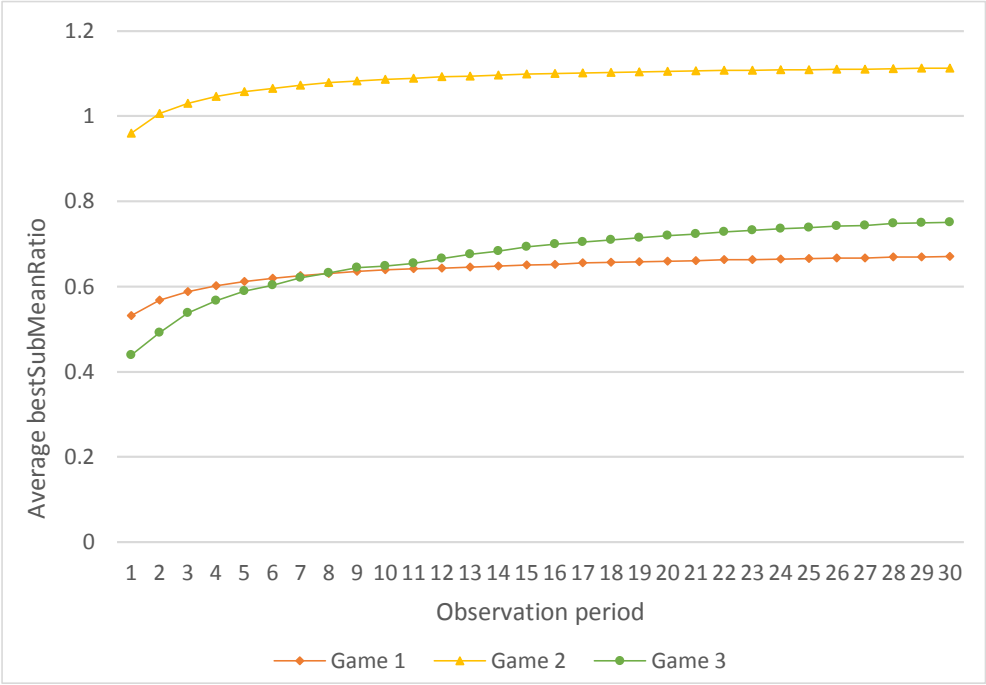

그림 44 관찰 기간 별 bestSubMeanRatio의 변화

위 그래프를 살펴보면 세 개의 게임 모두 증가하는 그래프로 나타났으며, 관찰 기간 5일 이후부터 수렴하는 형태를 보여주고 있다. 전체적으로 증가하는 그래프이기 때문에 유저들의 최고 점수가 평소 실력보다 점점 올라가고 있다고 볼 수 있고 이는 유저들이 지속적으로 학습을 통해서 지속적으로 최고 점수를 득점하고 있다고 해석할 수 있다. 또한, 관찰기간이 길어지면서 수렴하고 있는데 이는 유저들이 어느 순간부터 학습으로도 더 좋은 최고 득점을 기록하지 못하는 것으로 해석할 수 있다. 또한, Game 3의 전체적인 상승폭이 나머지 Game 1, Game 2보다 좋은데, 이는 Game 3의 게임 디자인 설계가 유저들이 학습을 통해서 더 좋은 점수를 만들 수 있는 정도가 나머지 게임에 비해서 크다라고 볼 수 있다. 또한, Game 2의 수치들이 나머지 수치들보다 높은데 이는 앞서 설명한 sdScore Feature에서 보았듯이 Game 2의 점수들끼리의 표준 편차가 큰걸 설명해주는 다른 근거가 된다고 해석 가능하다.

bestSubMeanCount

bestSubMeanCount는 적은 플레이로 높은 점수를 내는 유저를 알아내기 위한 Feature이다. 유저 별로 게임 내에서 기록한 점수들 중 최고 점수에 평균 점수를 뺀 값을 플레이 횟수로 나눈값을 bestSubMeanCount로 추출한다. bestSubMeanCount의 공식은 아래와 같다[공식 7].

$$bestSubMeanCount = \frac{Max(score) - \overline{score}}{n}$$

공식 7 Feature 산출 공식: bestSubMeanCount

위의 공식에서와 같이 적은 플레이로 최고 점수를 기록한 유저일수록 높은 bestSubMeanCount 값을 가진다. 또한, 같은 최고 점수와 평균 점수를 가진 유저라도 게임 플레이가 많을 수록 본 bestSubMeanCount값은 낮아진다. [그림 45]에서 전처리된 테이블에서 bestSubMeanCount를 추출하는 것에 대해 도식화하여 나타내었다

User A

score	time
4	2015-10-29 23:56:55
252	2015-10-30 00:08:11
169	2015-10-30 00:10:08
238	2015-10-30 00:15:01
184	2015-10-30 00:15:48
208	2015-10-30 00:19:38
353	2015-10-30 00:33:23
586	2015-10-30 07:48:13
472	2015-10-30 09:45:21

$\frac{Max(score) - \overline{score}}{n} = \frac{586 - 274}{9} \fallingdotseq 34.667$

User B

score	time
283	2015-05-24 16:00:11
469	2015-05-24 16:01:22
234	2015-05-24 16:13:34
338	2015-05-25 09:31:43
361	2015-05-27 14:44:19
297	2015-05-27 22:14:58
306	2015-05-27 22:16:31
243	2015-05-28 18:33:32
474	2015-06-03 13:32:17
413	2015-06-03 15:42:55

$\frac{Max(score) - \overline{score}}{n} = \frac{474 - 341.8}{10} = 13.22$

그림 45 Feature 정의: bestSubMeanCount

위에서 정의한 bestSubMeanRatio 를 기반으로 게임 별 전체 유저들을 대상으로 관찰 기간 별 평균 bestSubMeanRatio 의 변화에

대한 그래프를 [그림 46]와 같이 그려보았다

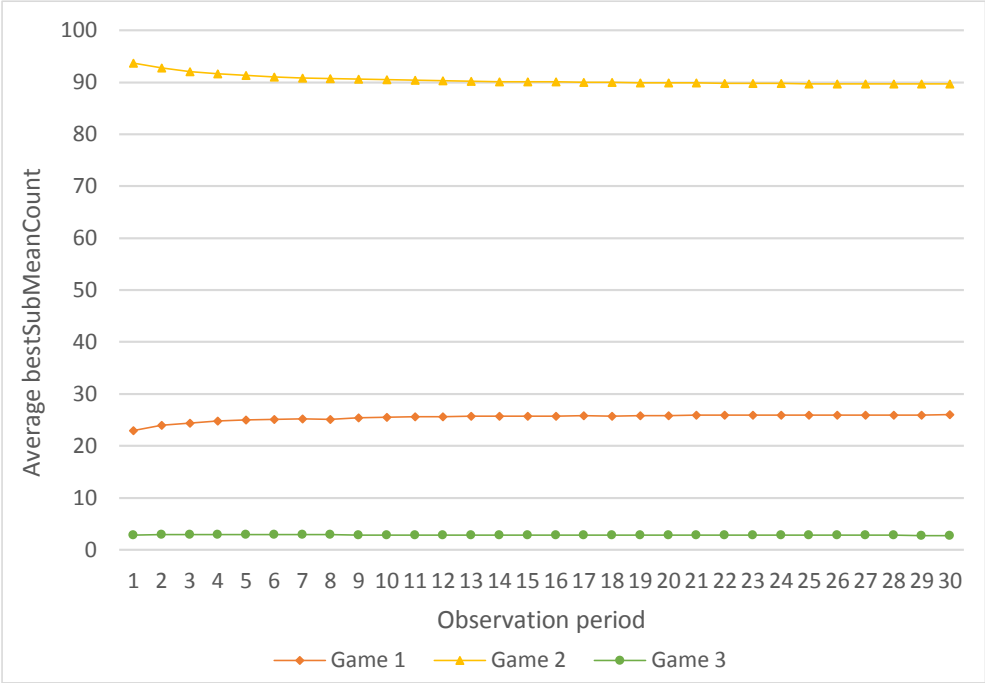

그림 46 관찰 기간 별 bestSubMeanCount의 변화

위 그래프에서 세 게임 모두 평균 bestSubMeanCount값의 변화가 적다. Game 2의 경우 소량 감소하는 그래프가 나왔다. 관찰 기간 첫 3일 동안 Game 2 평균 bestSubMeanCount값의 감소가 눈에 띄는데, 이는 관찰 기간이 길어질수록 적은 플레이로 높은 점수를 기록하는 유저들이 없어진다는 해석이 가능하다. 또한, Game 1은 처음 5일 동안의 관찰 간동안 소량 증가하는 그래프의 형태를 나타내고 있다. 이는 유저들이 첫 플레이를 하고 5일 동안은 적은 플레이로 높은 점수를 기록을 할 수 있었고 5일 이후 기간동안 이런 bestSubMeanCount값들이 플레이가 많아지고 더 이상의 큰 점수 획득이 없어 증가량이 줄어 들었다고 해석이 가능하다.

bestScoreIndex

bestScoreIndex는 관찰 기간 동안 전체 플레이 중 최고 점수를

언제 기록을 하였는지에 대한 Feature이다. 최고 점수를 기록한 시점이 몇 번째인지를 나타내는 값을 전체 플레이수로 나눈 값을 bestScoreIndex로 추출한다. bestScoreIndex값이 0에 가까울수록 전체 게임 플레이 중 게임 초반 플레이에 기록한 것이고, 1에 가까울수록 후반 플레이에 기록한 것이다. bestScoreIndex의 공식은 아래와 같다[공식 8].

$$bestScoreIndex = \frac{Index(Max(score))}{n}$$

공식 8 Feature 산출 공식: bestScoreIndex

[그림 47]에서 전처리된 테이블에서 bestScoreIndex를 추출하는 것에 대해 도식화하여 나타내었다

User A

score	time
4	2015-10-29 23:56:55
252	2015-10-30 00:08:11
169	2015-10-30 00:10:08
238	2015-10-30 00:15:01
184	2015-10-30 00:15:48
208	2015-10-30 00:19:38
353	2015-10-30 00:33:23
586	2015-10-30 07:48:13
472	2015-10-30 09:45:21

$\frac{Index(Max(score))}{n} = \frac{8}{9} \fallingdotseq 0.889$

User B

score	time
283	2015-05-24 16:00:11
469	2015-05-24 16:01:22
234	2015-05-24 16:13:34
338	2015-05-25 09:31:43
361	2015-05-27 14:44:19
297	2015-05-27 22:14:58
306	2015-05-27 22:16:31
243	2015-05-28 18:33:32
474	2015-06-03 13:32:17
413	2015-06-03 15:42:55

$\frac{Index(Max(score))}{n} = \frac{9}{10} = 0.9$

그림 47 Feature 정의: bestScoreIndex

User A의 586점이 최고 기록이고 이는 시간 순으로 정렬된 유저 테이블에서 위에서 8번째의 기록이기 때문에 9로 나눈 약 0.889라는 값이 나오게되었다.

위에서 정의한 bestScoreIndex를 기반으로 게임 별 전체 유저들을 대상으로 관찰 기간 별 평균 bestScoreIndex의 변화에 대한 그래프를 [그림 48]와 같이 그려보았다

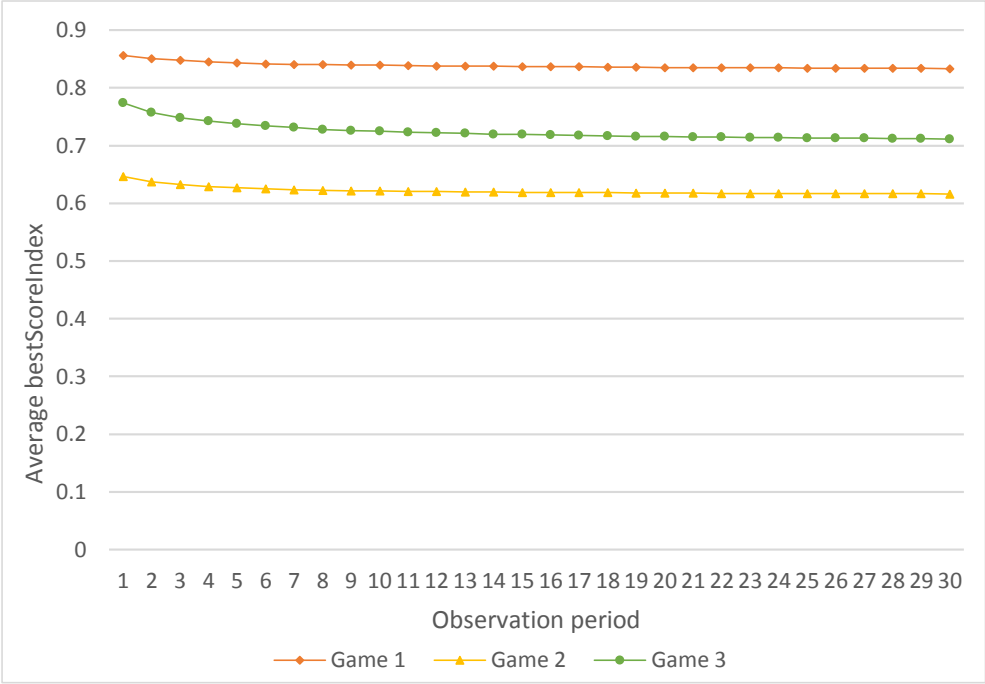

그림 48 관찰 기간 별 bestScoreIndex의 변화

위 그래프에서는 세개의 게임 모두 평균bestScoreIndex값이 0.6 이상이였고 관찰 기간이 길어질수록 감소하는 그래프로 나타났다. 이는 유저들의 최고 기록이 전체 플레이 중에 끝 쪽에 있다고 해석이 가능하다. 또한, 관찰 기간이 길어질수록 평균적으로 유저들이 게임 플레이를 할수록 대부분 최고 점수를 뛰어넘는 플레이를 하지 못하는 것으로 해석이 가능하다.

activeDuration

activeDuration은 관찰 기간 동안 유저가 얼마 기간동안 플레이를 하였는지에 대한 Feature이다. 마지막 게임 플레이 시각에 첫 게임 플레이 시각을 뺀 값을 activeDuration값으로 사용한다. activeDuration의 공식은 아래와 같다[공식 9].

$$activeDuration = Max(time) - Min(time)$$

공식 9 Feature 산출 공식: activeDuration

[그림 49]에서 전처리된 테이블에서 activeDuration를 추출하는 것에 대해 도식화하여 나타내었다

User A

score	time
4	2015-10-29 23:56:55
252	2015-10-30 00:08:11
169	2015-10-30 00:10:08
238	2015-10-30 00:15:01
184	2015-10-30 00:15:48
208	2015-10-30 00:19:38
353	2015-10-30 00:33:23
586	2015-10-30 07:48:13
472	2015-10-30 09:45:21

$Max(time) - Min(time)$
$= 1446198321 - 1446163015 = 35306$

User B

score	time
283	2015-05-24 16:00:11
469	2015-05-24 16:01:22
234	2015-05-24 16:13:34
338	2015-05-25 09:31:43
361	2015-05-27 14:44:19
297	2015-05-27 22:14:58
306	2015-05-27 22:16:31
243	2015-05-28 18:33:32
474	2015-06-03 13:32:17
413	2015-06-03 15:42:55

$Max(time) - Min(time)$
$= 1433346175 - 1432483211 = 862964$

그림 49 Feature 정의: activeDuration

위에서 정의한 activeDuration를 기반으로 게임 별 전체 유저들을 대상으로 관찰 기간 별 평균 activeDuration의 변화에 대한 그래프를 [그림 50]와 같이 그려보았다

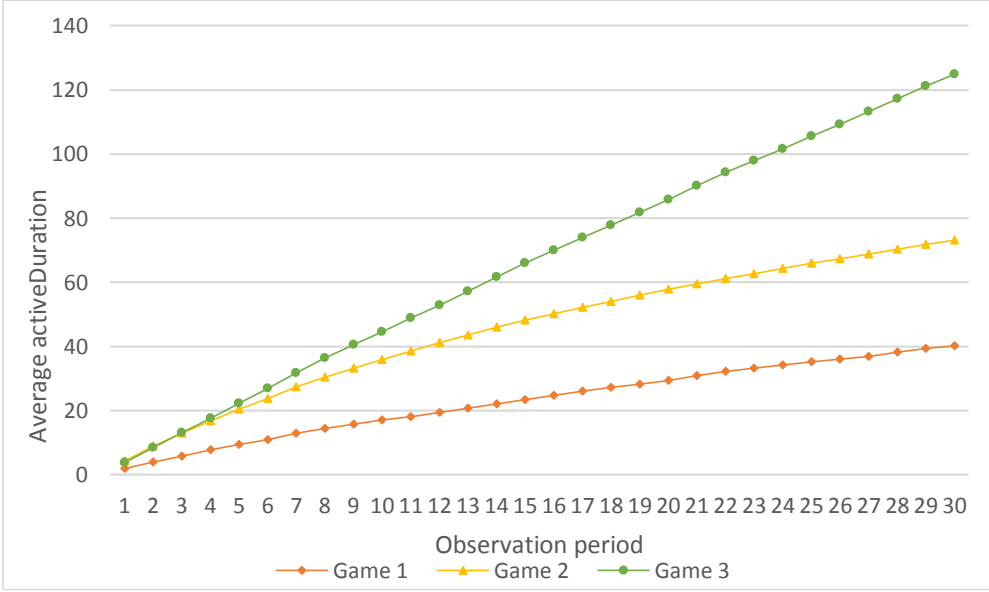

그림 50 관찰 기간 별 activeDuration의 변화

위 그래프는 세 개의 게임 모두 상승 곡선을 그려주고 있으며 Game 2가 Game1, Game 2에 비해 수렴하는 정도가 크다. 전체적인 유저의 평균 activeDuration은 길이는 Game 3가 가장 길고 Game 2, Game 1순으로 길이가 짧아진다. 또한, Game 2와 Game 3의 평균 activeDuration이 관찰 기간 초반 3일 동안은 비슷했으나 관찰 기간이 길어짐에 따라서 점점 차이가 났다. 이는 앞서 말했던 이탈률과도 관계가 있는데 Game 2와 Game 3의 초반 이탈율은 비슷했던 것에 비해 관찰기간 30일의 이탈율은 약 15%나 차이가 난 것을 설명해주는 또 다른 근거로 해석이 가능하다[그림 19].

2. 전용 Feature: Game 2

purchaseCount

purchaseCount는 탈것을 몇번 업그레이드 하였는지에 대한 Feature이다. 해당 관찰 기간 내에서 유저의 전체 탈 것 구매 내역의 행의 수를 purchaseCount로 추출한다. purchaseCount의 공식은 아래와 같다[공식 10].

$$purchaseCount = n$$

공식 10 Feature 산출 공식: purchaseCount

[그림 51]에서 전처리된 테이블에서 purchaseCount를 추출하는 것에 대해 도식화하여 나타내었다

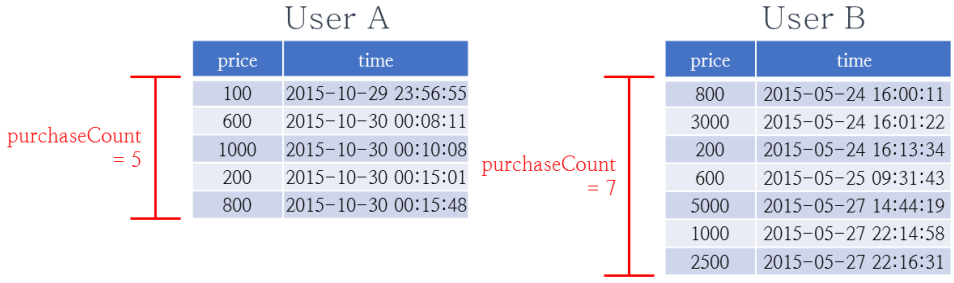

그림 51 Feature 정의: purchaseCount

위에서 정의한 bestSubMeanRatio 를 기반으로 게임 별 전체 유저들을 대상으로 관찰 기간 별 평균 purchaseCount의 변화에 대한 그래프를 [그림 52]와 같이 그려보았다

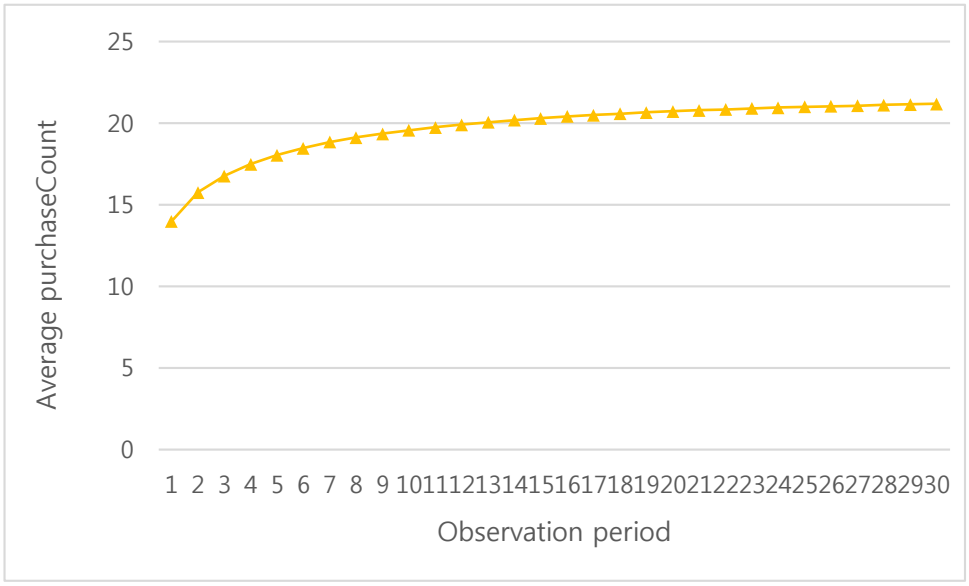

그림 52 관찰 기간 별 purchaseCount의 변화

위 그래프에서는 관찰 기간이 길어질수록 평균 purchaseCount는 증가되면서 수렴을 한다. 유저들은 관찰 기간 첫 일주일 동안은 구매를 점점 많이하다가 유저가 이탈하면서 전체적으로 점점 구매하는 횟수가 줄어 들었다고 볼 수 있다. 또한, 게임 디자인 상 게임 초반엔 한번 구매를 하는데 걸리는 시간이 적고, 비용도 작지만 게임을 재미있게 플레이를 하고 난 이후로는 실제 과금을 유도하기 위해 상점에서 구매할 수 있는 가격을 급격히 높이고, 비용도 높게하여 과금을 하지 않을 수 없도록 디자인이 되어있는것도 관찰 기간의 증가에 따른 평균 purchaseCount과 이탈률에 큰 역할을 할 수 있다. 이는 과금 시점에 대한 추가적인 연구의 필요성을 보여주고 있다.

bestPurchase

bestPurchase는 상점에서 구매한 내역 중 가장 비싸게 주고 산

상품의 가격에 대한 Feature이다. 해당 관찰 기간 내에서 유저의 구매 내역 중 가장 큰 수치를 bestPurchase로 추출한다. bestPurchase의 공식은 아래와 같다[공식 11].

$$bestPurchase = Max(price)$$

공식 11 Feature 산출 공식: bestPurchase

[그림 53]에서 전처리된 테이블에서 bestPurchase를 추출하는 것에 대해 도식화하여 나타내었다

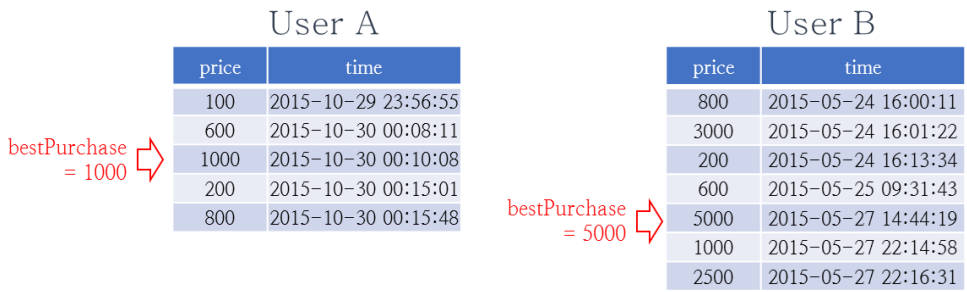

그림 53 Feature 정의: bestPurchase

위에서 정의한 bestPurchase를 기반으로 게임 별 전체 유저들을 대상으로 관찰 기간 별 평균 bestPurchase의 변화에 대한 그래프를 [그림 54]와 같이 그려보았다

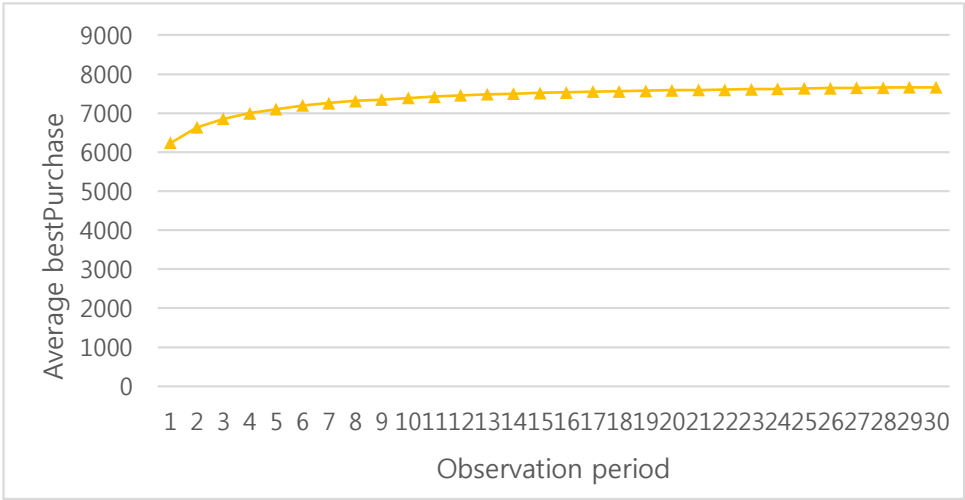

그림 54 관찰 기간 별 bestPurchase의 변화

위 그래프도 앞서 설명한 purchaseCount와 같은 증가 형태를 보이고 있다. 관찰 기간이 길어질수록 유저의 평균 최고 금액이 증가하지만 일주일이 넘어가면서 이탈하는 유저들의 많아지고 상점에서 구매하는 횟수도 줄어들기 때문에 자연히 유저들의 평균 bestPurchase 값도 낮아 진다는 해석이 가능하다.

3. 전용 Feature: Game 3

winRatio

winRatio는 유저의 승률을 나타내는 Feature이다. 해당 관찰 기간 내의 point 데이터 중 0이 아닌 값의 개수를 전체 데이터 수로 나눈 수이다. point값은 유저가 승리를 할 때 획득하는 점수로 유저가 패배를하게되면 점수를 전혀 얻지를 못한다. winRatio의 공식은 아래와 같다[공식 12].

$$winRatio = \frac{nrow(point>0)}{n}$$

공식 12 Feature 산출 공식: winRatio

[그림 55]에서 전처리된 테이블에서 winRatio를 추출하는 것에 대해 도식화하여 나타내었다

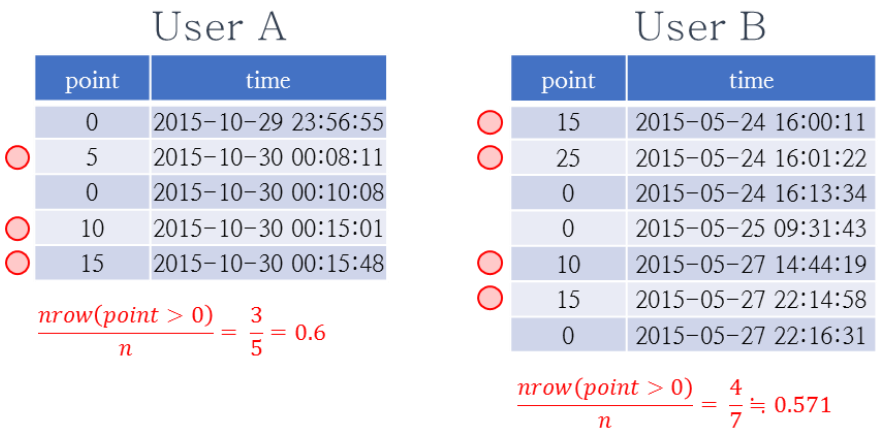

그림 55 Feature 정의: winRatio

위에서 정의한 winRatio를 기반으로 게임 별 전체 유저들을 대상으로 관찰 기간 별 평균 winRatio의 변화에 대한 그래프를 [그림 56]와 같이 그려보았다

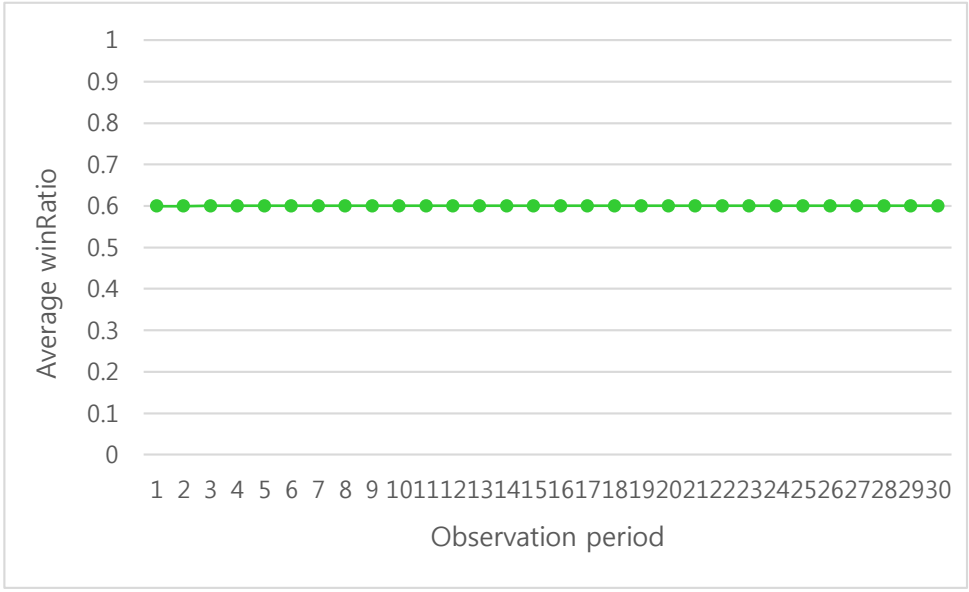

그림 56 관찰 기간 별 winRatio의 변화

위 그래프에서는 관찰 기간 별 winRatio의 변화가 거의 없다. 또한, 평균 승률이 약 60%로 나오는데 이는 게임 디자인 상황이 반영된 수치라고 볼 수 있다. Game 3에서는 게임 플레이시 팀을 두개로 나눠서 진행을 하는데 팀이 제한 시간이 모두 지났고 팀 점수가 동점일 경우 두 팀 모두 승리한 것으로 보고 승리에 대한 point를 획득하게 해준다. 단, 플레이 도중에 이탈 한 유저들에 한해서는 point 획득이 불가하다. 아래 [표 13]은 Game 3의 승률에 대한 전체 통계이다.

	Red	Blue
팀 별 승률 (동점포함)	63%	65%
팀 별 승률 (동점제외)	49%	51%

표 13 Game 3의 팀 별 전체승률

위 표를 보면 알 수 있듯이 평균 승률이 약 60%가 나온 것은 동점이라는 개념과 플레이 도중 이탈 유저에 대한 처리와 같은 게임 디자인 상의 이유로 볼 수 있다.

gameDurationMean

gameDurationMean은 유저가 플레이한 게임들의 평균 게임 시간을 뜻하는 Feature이다. 전처리한 Game 3의 시간들의 평균 수치를 gameDurationMean으로 추출한다. gameDurationMean의 공식은 아래와 같다[공식 13].

$$gameDurationMean = \frac{\sum_{i=1}^{n} duration_i}{n}$$

공식 13 Feature 산출 공식: gameDurationMean

[그림 57]에서 전처리된 테이블에서 gameDurationMean를 추출하는 것에 대해 도식화하여 나타내었다

User A

duration	time
16356	2015-10-29 23:56:55
13359	2015-10-30 00:08:11
24788	2015-10-30 00:10:08
24285	2015-10-30 00:15:01
19813	2015-10-30 00:15:48

$\frac{\sum_{i=1}^{n} duration_i}{n} = \frac{98601}{5} \fallingdotseq 19720$

User B

duration	time
31821	2015-05-24 16:00:11
26256	2015-05-24 16:01:22
12197	2015-05-24 16:13:34
22624	2015-05-25 09:31:43
29300	2015-05-27 14:44:19
31764	2015-05-27 22:14:58
26664	2015-05-27 22:16:31

$\frac{\sum_{i=1}^{n} duration_i}{n} = \frac{180626}{7} \fallingdotseq 258033$

그림 57 Feature 정의: gameDurationMean

위에서 정의한 gameDurationMean를 기반으로 게임 별 전체 유저들을 대상으로 관찰 기간 별 평균 gameDurationMean의 변화에

대한 그래프를 [그림 58]와 같이 그려보았다

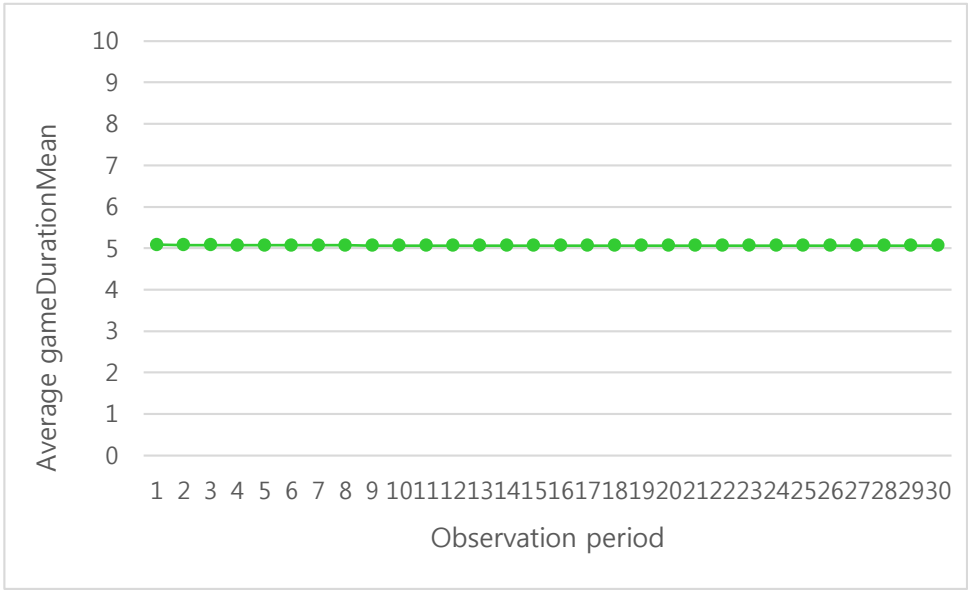

그림 58 관찰 기간 별 gameDurationMean의 변화

위 그래프에서는 유저들의 평균 gameDurationMean이다. 관찰 기간과 상관없이 5분을 유지하고 있다. 아래 [그림 59]는 gameDuration의 히스토그램을 나타내고 있다.

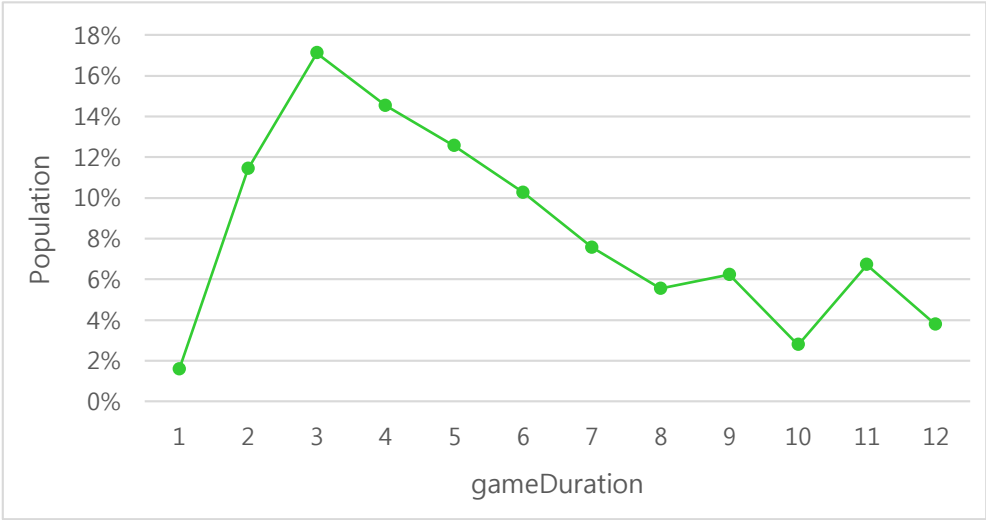

그림 59 gameDuration 별 Population

게임이 끝나는 시간대가 3분이 가장 많았고 3분 이후로 점점

줄어들다가 11분에서 10분에 비해 두배 가까이 population이 올라간다. 이는 대부분의 게임이 3분에 게임이 끝나고 그 이후 시간동안 어느정도 게임 결과가 결정이 나지만, 게임 제한 시간이 얼마 남지 않은 마지막 11분에서 게임의 반전을 노려보려 집중을해서 게임을 끝내는 경우가 많다라고 해석이 가능하다.

제 4 절 유저 이탈 예측 모델 설계

4절에서는 유저 이탈 예측에 사용될 3가지 모델에 대해서 설계를 하도록 한다. Gradient boosting, Logistic regression, Random forest를 이용한 알고리즘에 대한 설명과 동시에 이들 알고리즘을 통해서 각각 유저 이탈 예측 모델을 설계한다.

1. Gradient boosting를 이용한 유저 이탈 예측 모델

Gradient boosting은 deision tree와 같은 약한 예측 모델들의 ensemble 형태를 가지고 regression 및 classification을 하기 위한 강한 예측 모델을 만드는 머신러닝 기법이다 [33]. 그리고 다른 알고리즘에 비해 경쟁력이 있으며, 확실성, 해석성이 뛰어나다는 장점이 있다[34]. 또한, Gradient boosting은 Classification tree analysis의 가장 최적의 tree를 찾지 못하고, outlier, 부정확한 training data, 불균형한 데이터 셋의 한계들을 해결해 준다[35]. 이러한 Gradient boosting 모델은 classification error를 최소화하는 방향으로 training data를 랜덤으로 샘플링하여 반복적으로 분류기를 생성하여 결합하는 방식으로 데이터를 분류한다[36].

Gradient boosting 알고리즘에 대한 설명은 [공식 14]에서부터 유도가 된다.

$$F_{m+1}(x) = F_m(x) + h(x) = y,$$
$$h(x) = y - \ F_m(x),$$

공식 14 Gradient boosting 유도 공식 1

Gradient boosting의 $1 \le m \le M$ 의 단계에서 $F_m(x)$ 을 약한 분류기 혹은 모델로 정의를 한다. Gradient boosting 알고리즘은 $F_m(x)$ 자체를 변경하지는 못하고 추정값 h를 통해서 더 나은 $F_{m+1}(x)$ 모델을 생성한다[37].

$$F_0(x) = arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, \gamma),$$

$$F_m(x) = F_{m-1}(x) + arg\min_{f \in \mathcal{H}} \sum_{i=1}^{n} L(y_i, F_{m-1}(x_i + f(x_i)),$$

공식 15 Gradient boosting 유도 공식 2

[공식 15]은 empirical risk minimization의 기초에 따른 공식으로써, 손실함수 L의 평균값을 최소화하는 근사치 $\hat{F}(x)$를 찾고자 한다[38]. $F_0(x)$ 값은 상수값으로써본 함수로부터 greedy 방식으로 연산이 시작된다. f는 약간 예측 모델이고 이런 모델들의 집합인 $\mathcal{H}$에 속해있다. 이렇게 m 값을 증가시켜서 greedy한 방식으로 $\hat{F}(x)$찾아야하는데, 손실함수L에 대한 가장 최적의 f값을 선택해야 하는 문제가 있다. 그래서 이를 Gradient descent방식으로 최적의 값을 찾아낸다[공식 16].

$$F_m(x) = F_{m-1}(x) - \gamma_m \sum_{i=1}^{n} \nabla_f L(y_i, F_{m-1}(x_i)),$$

$$\gamma_m = arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) - \gamma \frac{\partial L(y_i, F_{m-1}(x_i))}{\partial f(x_i)}.$$

공식 16 Gradient boosting algorithm 공식

본 연구에서도 Gradient boosting을 이용하여 유저 이탈 예측 모델을 만들도록 한다. 앞서 정의한 유저 별 이탈 분류와, 정의된 공통 Feature 10가지, 전용 Feature 4가지들을 가지고 플레이 기간을 기준으로 Train set, Test set으로 나누도록 한다. [그림 60]에 Train set, 과 Test set으로 나누는 것을 도식화하여 설명하였다.

isChurn	playCount	bestScore	consePlay	worstScore	meanScore	sdScore	bestSubMeanRatio	bestSubMeanCount	bestScoreIndex	activeDuration
0	141	699	0.871	0	201	157	2.462	3.526	0.411	334504
0	1	1336	0.000	1336	1336	0	0.000	0.000	1.000	0
0	115	1666	0.921	1	409	356	3.067	10.924	0.600	305706
⋮										
1	18	22	0.705	0	7	5	2.246	0.845	0.277	246991
1	3	147	1.000	69	113	39	0.305	11.444	1.000	169
0	15	115	0.928	1	42	29	1.729	4.854	0.867	42255
1	1	10	0.000	10	10	0	0.000	0	1.000	0

Training data

Test data

그림 60 Training, Test data 나누기

위와 같이 나뉘어진 Traning data를 가지고 이탈 예측 모델을 생성하고 Test data를 통해서 예측 성능을 평가한다.

2. Logistic regression를 이용한 유저 이탈 예측 모델

회귀분석을 통해 독립변수와 종속변수와의 관계를 요약해볼 수 는데, 가장 일반적인 방법은 Linear regression이다[39]. 그러나 Linear regression은 본 연구와 같이 종속변수가 이진형으로 나누어진 값을 가질 때에는 적용하는 데에 문제가 생긴다. [그림 61]를 보면 동일한 데이터를 가지고 Linear regression과 Logistic regression 그래프를 비교하였다[40].

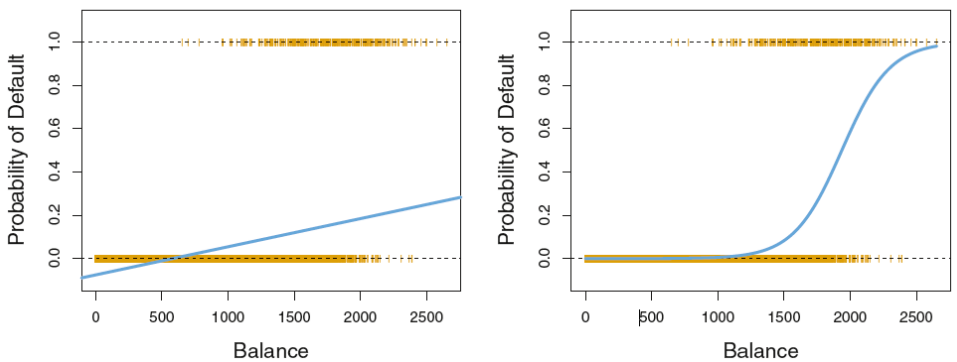

그림 61 Linear regression(좌측), Logistic regression우측)의 예시

좌측의 Linear regression로 이진값을 분류한다면 몇몇의 값은 음수로 분류된다. 위와 같은 문제로 인해서 본 연구에서는 이진형 분류에 특화된 Logistic regression를 이용하여 모델 설계를 하려고 한다.

종속 변수가 이진형인 Logistic regression의 경우 독립변수의

값이 증가함에 따라 종속변수의 값이 0의값부터 점차 증가하여 1에 수렴하는 형태를 띠게 된다. 이를 식으로 표현하면 [공식 17]과 같다.

$$p(X) = \frac{e^{B_0+B_1X}}{1+e^{B_0+B_1X}}$$

공식 17 Logistic gression 공식

본 연구에서 사용할 데이터를 [그림 62]에서와 같이 Training set을 가지고 플레이횟수를 Feature로 삼아 Logistic regression를 통해 그려보면 [그림 57]과 같다.

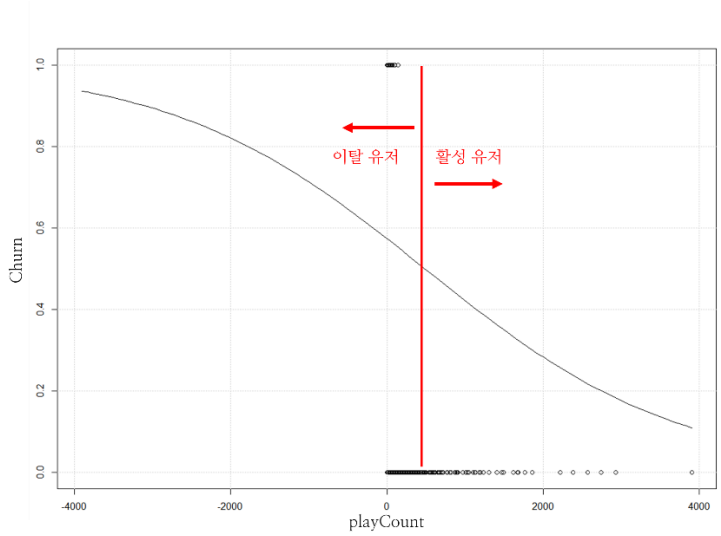

그림 62 플레이횟수에 대한 Logistic regression

위 그림에서와 같이 Logistic regression 함수를 통해 그린 곡선으로 인해 이탈 유저와 활성유저가 나뉘어지게 된다. 플레이 횟수가 적을수록 이탈 확률이 올라가고 플레이 횟수가 높을수록 이탈 확률이 줄어들게 된다.

3. Random forest를 이용한 유저 이탈 예측 모델

Random forest는 머신러닝에서 classification, regression에 사용되는 ensemble 방법 중에 하나로 training data를 가지고 다수의

decision tree부터 classification 혹은 regression을 하면서 작동이 되는 알고리즘이다[41, 42]. 이 알고리즘은 Breiman(2001)의 논문에서 randomized node optimization과 bagging을 결합한 CART(classification and regression tree)를 사용해 상관관계가 없는 트리들로 포레스트를 구성하는 방법을 제시하였다[42, 43]. 일반적인 Decision tree는 불규칙한 패턴을 넣었을때 tree깊이가 깊어질 경우는 불규칙한 패턴으로 training set에 대해 overfit하는 경향이 생기는데 반해, Random forest는 다양한 tree를 종합하고 같은 training set에 대해 다른 부분들 train을 하기 때문에 overfit 문제해결이 가능하다[44].

일반적인 Random forest는 ensemble기법인 bagging과 randomized node optimization을 통해서 예측을 하게 된다. 다음 [그림 63 Random forest 도식화]에 일반적인 Random forest 알고리즘을 도식화하였다.

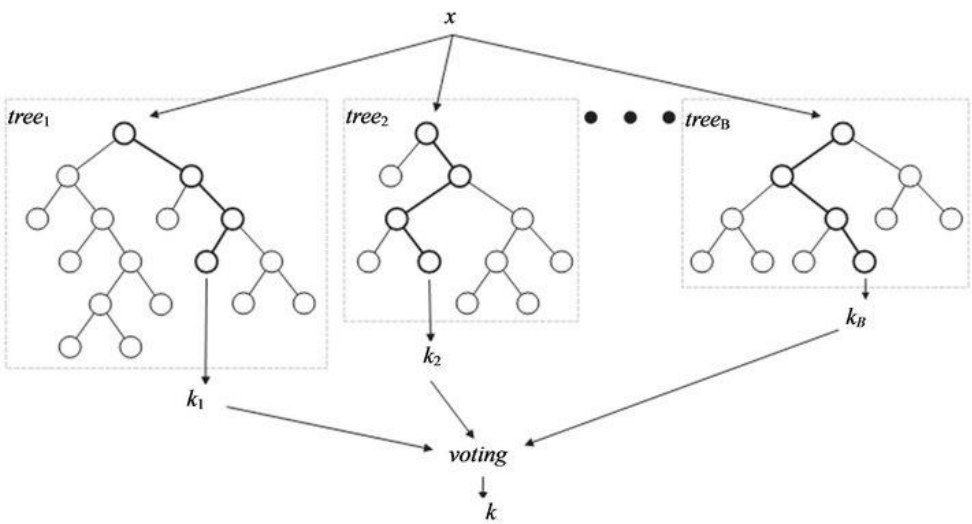

그림 63 Random forest 도식화

본 연구에서도 Random forest 알고리즘을 사용하여, 주어진 관찰 기간 및 이탈 예측 기간 별로 만든 게임별 데이터 테이블을 가지고 여러 개의 tree들을 생성하고 이 tree들에 대해 bagging과 randomized node optimization을 통해 이탈 예측 모델을 만들고 성능 평가를 하도록 한다.

제 5 장 실험 및 성능 평가

앞서 4장에서는 유저 이탈에 대해서 정의하였고 데이터 전처리 과정을 통해서 서버에서 가져온 데이터를 분석하기 용이하게 가공하였다. 또한, 예측 모델에 Input으로 집어넣을 Feature를 정의하였고 알고리즘별로 모델을 설계하였다.

5장은 실제 실험을 하고 성능 평가를 하도록 한다. 1절에서는 기존의 데이터를 알고리즘별로 설계한 유저 이탈 예측 모델에 인풋으로 넣을 수 있는 데이터로 변환한다. 2절에서는 실제 실험을 통한 각 모델별의 실험 결과를 10-fold Cross Validation을 통한 ROC(Receiver operating characteristic) curve의 AUC(Area under the curve)을 통해서 보여주도록 한다. 3절에서는 예측 모델과 Random Guessing의 AURPC(Area under the Precision Recall Curve)의 비교를 통해서 예측 모델별, Feature별 성능 평가를 하도록 한다.

제 1 절 데이터 변환

데이터를 예측하는 데 있어서 데이터 변환 단계는 매우 중요한 단계이다. 우선, 예측(Prediction)에 관해서 설명하자면, 다음과 같다[공식 18].

$$\check{Y} = \hat{f}(X)$$

공식 18 Prediction function

예측하는 데 있어서 Input X는 손쉽게 구할 수 있지만, Output Y는 손쉽게 얻을 수 없다. 위의 공식은 에러가 없을 때의 Input X에 대한 Output Y의 공식이다. $\hat{f}$는 f를 추정하는 함수이다. $\check{Y}$는 Y를 예측하려는 결과값이다. 위 공식에서 $\hat{f}$는 실제로 정확한 $\hat{f}$의 형태를 가지고 있지 않기 때문에 보통 black box로 불린다[38]. 위와 같은 예측 함수에 Input X에 집어넣기 위해서는 Input X가 함수에 맞게 변환이 되어있어야 한다. 알맞게 변환이 되지 않은 상태의 Input X은

제대로 된 결과를 주지 못한다. 그러므로 데이터 변환 단계가 필요하다.

데이터 변환 단계에서는 데이터 전처리 단계에서 가공한 데이터를 가지고 예측 모델에 넣기 위한 변환을 시작한다. 전처리 되어있는 데이터는 [표 14, 표 15, 표 16]와과 같다.

Game 1: basic table			
id	device	score	time
56124	255123547523145	4363	1406267278
56125	255123547523145	334	1406267280
56126	884152334561122	221	1406267281
56127	884152334561122	446	1406267290

표 14 Game 1 전처리 된 데이터

Game 2: score table			
id	device	score	time
56124	63BFD298E47F9010129DAB28C9390E	4363	1406267278
56125	63BFD298E47F9010129DAB28C9390E	334	1406267280
56126	E47F9010645FA67BBD8C938C938C93	221	1406267281
56127	E47F9010645FA67BBD8C938C938C93	446	1406267290

Game 2: price table			
id	device	price	time
74162	A7E0855A49F0F28E955748035008D6	500	1406263428
74163	A7E0855A49F0F28E955748035008D6	800	1406263430
74164	CE26DEF6B3305E9554748E55E20463	3000	1406263452
74165	CE26DEF6B3305E9554748E55E20463	5500	1406263480

표 15 Game 2 전처리 된 데이터

Game 3: basic table					
id	device	score	point	duration	time
24882	ksw29zz	500	15	24685	1406651521
24883	ksw29zz	800	0	15221	1406663728
24884	sunnyjun	3000	25	31186	1406683515
24885	sunnyjun	5500	0	42215	1406691521

표 16 Game 3 전처리 된 데이터

위의 표와 같이 세 게임에 대한 전처리 된 데이터는 모두 id 값을 가지고 있고, 고유의 device 값을 가지고 있으며, unix time으로 변환된 time 값을 가지고 있다. 또한, 전처리된 데이터는 시간 순으로 오름차순으로 정렬이 되어있다.

전처리된 테이블을 가지고 데이터 변환 단계를 통해서 유저이탈 예측 모델에 Input으로 넣을 데이터 테이블로 변환을시켜야 한다. 변환이 완료된 데이터 테이블의 구조는 [그림 64]과 같다.

device	isChurn	playCount	bestScore	consecutive PlayRatio	worstScore	meanScore	...	active Duration
2	0	141	699	0.871	0	202		334504
9	0	1	1336	0.000	1336	1336		0
12	0	115	1666	0.921	1	410		305706
35	1	18	22	0.706	0	7		246991
64	1	3	147	1.000	69	113		169
67	0	15	115	0.929	1	42		42255
72	1	1	10	0.000	10	10		0
73	1	4	115	1.000	39	77		121
92	1	2	137	1.000	32	85		60
112	1	4	988	0.000	30	367		334678
116	1	2	1850	1.000	1402	1626		510
118	0	2	421	1.000	5	213		63

그림 64 변환이 완료된 데이터 테이블 예시

위 그림에서와 같이 한 device에대해 여러 개의 데이터가 있었던 전처리된 데이터 테이블을 하나의 데이터 행으로 변환한다. 또한, 빠른 컴퓨팅 속도를 위해 device이름을 고유 id로 변환한다.

isChurn 속성은 이탈의 여부를 가르키는 속성으로써 0일 경우엔 활성유저, 1일 경우엔 이탈 유저이다. isChurn 값을 추출 할 때엔 앞서 4장에서 설명한바와 같이 정해진 이탈예측 기간(Churn prediction period)에 플레이 기록이 없으면 이탈유저, 플레이 기록이 존재하면 활성유저로 분류를 한다. isChurn은 이탈예측 기간에 따라 값이 변할 수 있다.

playCount 이후의 속성은 앞서 정의한 Feature 속성들이다. Game 1은 공통된 10개의 Feature 열들이 있고, Game 2와 Game 3는 각각 공통된 10개의 Feature 열들과 전용 2개 Feature 열들, 총 12개씩의 Feature열들이 있다. Feature 속성은 관찰 기간에 따라서

값이 변할 수 있다.

앞서 설명 하였듯이 본 연구에서는 관찰 기간과 이탈예측 기간을 1일부터 30일 까지 변화해 가면서 예측 성능에 끼치는 영향을 보기로 한다. 위에서 변환한 데이터 테이블은 하나의 관찰 기간과 하나의 이탈예측 기간에 대한 테이블이다. 관찰 기간과 이탈예측 기간을 변화해 가면서 예측 성능을 보기 위해서는, 세개의 게임에 대하여 관찰 기간 30개와 이탈예측 기간 30개를 조합하여 총 2,700개의 데이터 테이블을 생성해야한다. [그림 65]에 변환된 테이블에 대한 연구의 흐름을 도식화하였다.

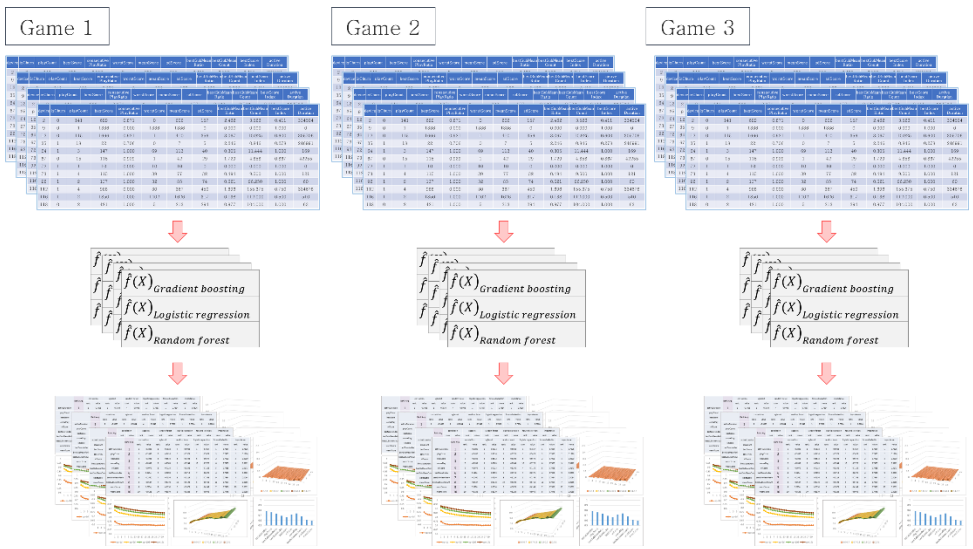

그림 65 변환된 테이블에 대한 연구의 흐름

위의 그림과 같이 변환된 2,700개의 테이블을 가지고 유저 이탈 예측 모델을 Gradient boosting, Logistic regression, Random forest알고리즘 별로 생성하고, 예측 결과에 대해서 여러 관점으로 분석을 해보기로 한다.

제 2 절 성능 평가

앞서 1절에서 데이터 변환과정을 통해 2,700개의 테이블을 생성하였다. 위 테이블을 가지고 알고리즘 별 예측 모델을 생성한다. 생성된 예측 모델의 예측 결과에 대한 예측 성능을 평가 해야하는데 본 2절에서는 예측 모델을 검증하는 방법인 10-fold Cross Validation과 예측 성능 평가에 쓰이는 ROC(Receiver operating characteristic)와 AUC(Area under the curve)에 대해서 살펴본다.

1. 10-fold Cross Validation

10-fold Cross Validation은 모델을 검증하는 테크닉 중 하나로 예측 모델이 각각의 독립적인 data set에 일반화되는지를 평가한다[45]. 2-fold Cross Validation, 20-flod Cross Validation이 있지만, 그 중 10-fold Cross Validation이 가장 보편적으로 사용된다[46]. [그림 66]에서 10-fold Cross Validation과정에 대해 도식화하였다.

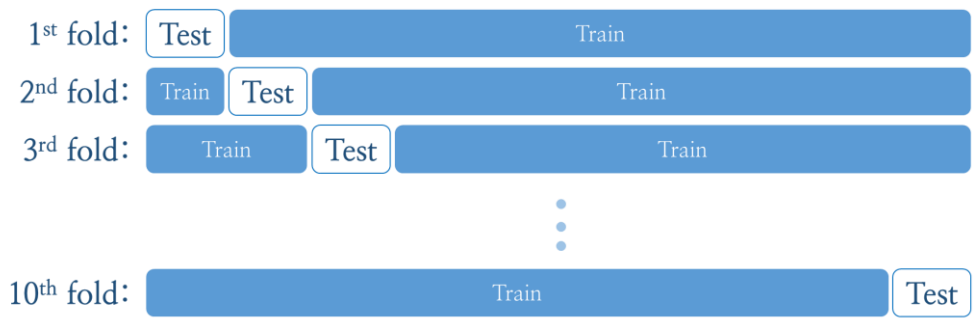

그림 66 10-fold Cross Validation

전체 data set을 10개로 나누어서 첫 번째 data set을 Test set으로, 나머지 9개 data set을 Train set으로 나누어서 성능을 본다. 그 다음으로 두 번째 data set을 Test set으로, 나머지 9개의 data set을 Train set으로 나누어서 성능을 본다. 이렇게 10번의 iteration을 돌고 10개의 성능을 통해 비교 및 분석을 한다.

2. ROC(Receiver operating characteristic)와 AUC(Area under the curve)

ROC curve는 classification의 성능을 평가에 쓰이는 그래프이다[47]. 이런 ROC curve는 분류와 시각화에 효과적이여서 의학 부분의 decision making부분에 많이 사용하였고 최근 머신러닝과 데이터 마이닝에서 많이 쓰이고 있다[48] 특히 ROC curve의 AUC는 성능 평가 부분에 아주 많이 쓰이는 방법이다[49]. 또한, ROC curve의 AUC는 간단하고 직관적으로 해석이 가능한 장점이 있다[50]

ROC curve를 그리기 위해서는 confusion matrix를 통해서 True Positive Rate(Sensitivity)와 False Positive Rate(1-Specifity)를 구해야 한다. 다음 [그림 67]은 Confusiopn matrix에 대한 도식화이다[51].

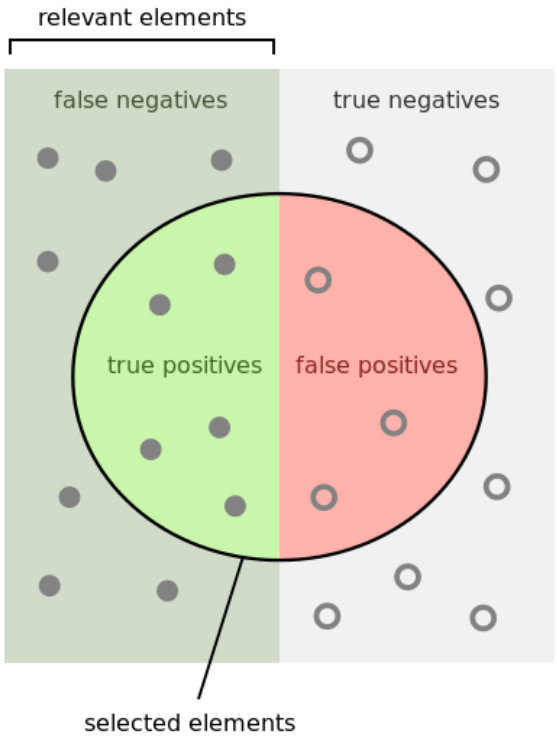

그림 67 Confusion matrix 도식화

위에서 True Positive는 실제 true인 것을 true라고 예측, Fasle Positive는 true라고 예측했는데 실제 fasle인 경우, False Negative는 실제 true인 것을 false라고 예측한 경우, True Negative는 fasle라고 예측했는데 실제 false인 경우를 말한다. 다음 [공식 19]은 ROC curve에서 사용 될 True Positive Rate와 False Positive Rate이다.

$$True\ \ Positive\ \ Rate = \frac{True\ \ Positives}{True\ \ Positives + False\ \ Negative}$$

$$False\ \ Positive\ \ Rate = \frac{False\ \ Positives}{False\ \ Positive + True\ \ Negatives}$$

공식 19 TPR, FPR 공식

위와 같이 True Posirive Rate와 False Positive Rate 계산을 하는데 다음과 같이 사용된다. 예측 된 테이블에서 Score를 높은 순으로 정렬하고 Threshold를 위에서 아래로 내려가면서 True Posirive Rate와 False Positive Rate를 계산한다. Threshold를 위에서 아래로 내려감에 따라 X축인 False Positive Rate이 점점 증가되면서 Y축 True Positive Rate에서 해당하는 값을 찾아서 좌표를 찍게 된다. 이 일련의 과정을 도식화하면 [그림 68]과 같다.

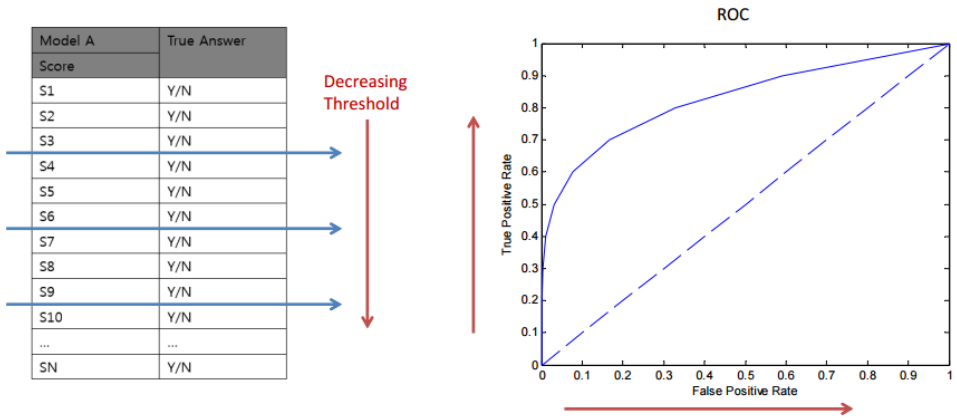

그림 68 ROC curve 그리기

예측 모델을가지고 그린 ROC curve의 AUC를 구하여 예측 모델의 성능을 평가를 하는 데 AUC의 범위는 0.5부터 1까지이고 세부적인 구간 범위에 대한 설명은 다음 [표 17]과 같다[52].

AUC Range	Interpretation
AUC = 0.5	Null model
0.5 < AUC < 0.7	Low accurarcy
0.7 <= AUC < 0.9	Moderate accuracy
0.9 <= AUC < 1	Hight accuracy
AUC = 1	Perfect accuracy

표 17 AUC 구간 별 해석

제 3 절 실험 결과

1절에서 변환한 데이터 테이블을 가지고 예측 모델을 알고리즘별로 생성을 하였고, 생성된 알고리즘을 2절에서 설명한 10-fold cross validation 검증 기법을 통해 ROC curve의 AUC 값으로 결과를 살펴보도록한다.

1. Single feature analysis

앞서 설명한 첫 번째 연구문제인, Feature들이 예측 성능에 얼마나 기여하는지를 보기 위해서 각기 Feature 별로 분석을 해보고 Feature의 순위를 구해본다. Guyon(2003)의 연구에 따르면 Feature의 순위를 구하기 위해서는 correlation coefficient을 구하는 방법, 하나의 Feature만 classification에 넣는 방법, 다른 Feature들과의 연관성을 보는 방법, 적당한 함수를 찾아서 구하는 방법이 있다[53]. 본 연구에서도 각 Feature 별로 유저 이탈과의 correlation coefficient, single Feature의 ROC curve의 AUC, forward selection의 결과, xgboost에서 제공하는 importance method를 통한 예측 모델의 기여도를 살펴보고 평균 순위를 내보도록 한다.

Correlation coefficient

세개의 게임 별로 각각의 Feature가 유저의 이탈에 얼마나 연관이 있는지 correlation coefficient을 구해보도록 한다. 이탈 예측 모델에서 사용되는 데이터 테이블에서 이탈 여부에 대한 열과 각 Feature 열들과의 correlation coefficient 을 구한다. correlation coefficient 은 Pearson correlation coefficient로 구하도록 한다. 또한, 비교를 위해서 correlation coefficient값는 절대값으로 치환하여 비교를 하였다.

아래 [그림 69]은 게임 별 Feature와 유저 이탈과의 correlation coefficient이다.

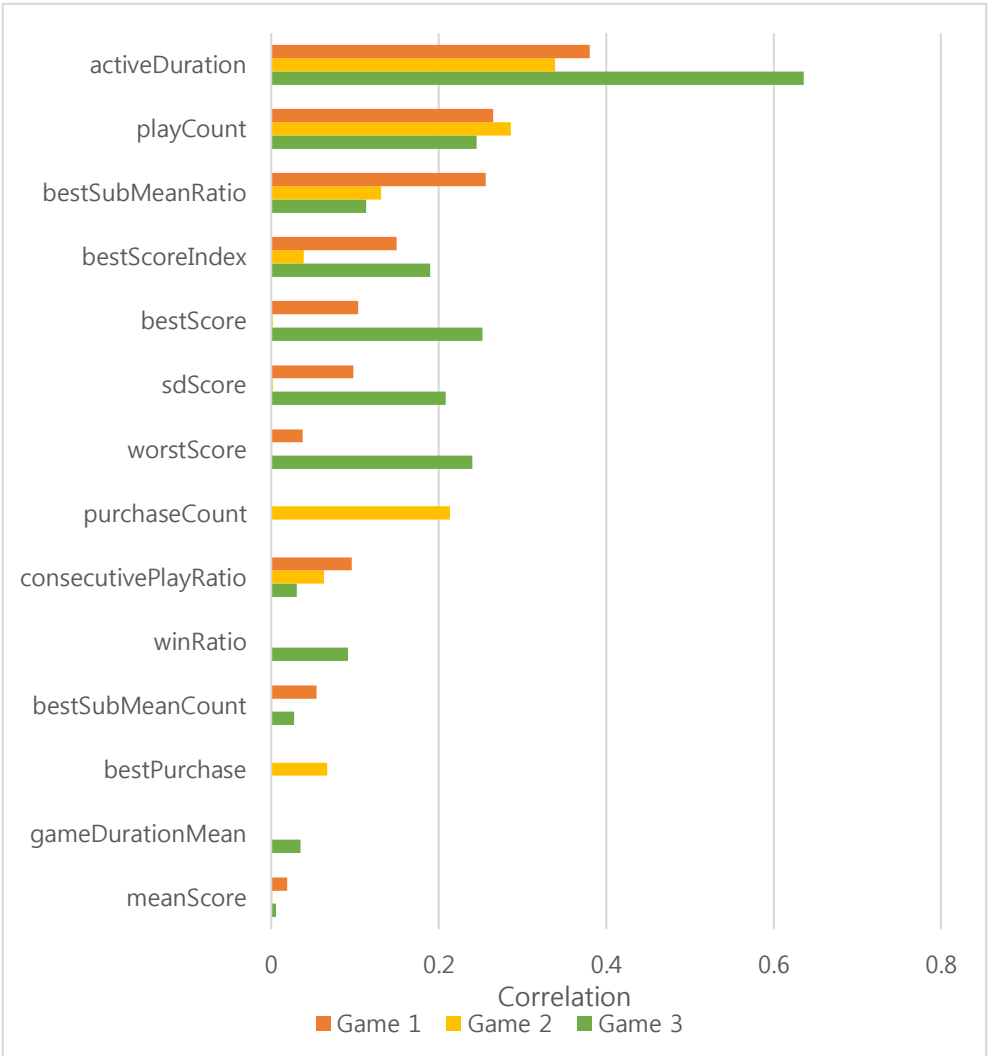

그림 69 게임 별 Feature와 유저 이탈과의 correlation

Game 3가 전체적으로 correlation coefficient결과가 높게 나왔다. 또한, activeDuration과 playCount가 세개의 게임 공통으로 높게 나왔다. 특정 게임에서만 correlation coefficient이 높에 나오는 Feature도 있는데 Game 1은 bestScoreIndex가 높게 나왔고, Game 2에서는 purchaseCount가 높게 나왔다. 그리고 Game 3에서는 bestScore, bestScoreIndex, sdScore, worstScore가 높게 나왔다. 또한, 세개의 게임 모두다 meanScore의 correlation coefficient 이 아주 낮게 나왔다.

Single feature AUC: Gradient boosting

아래의 [그림 70]은 Single feature를 Gradient boosting에 넣고 돌린 결과이다.

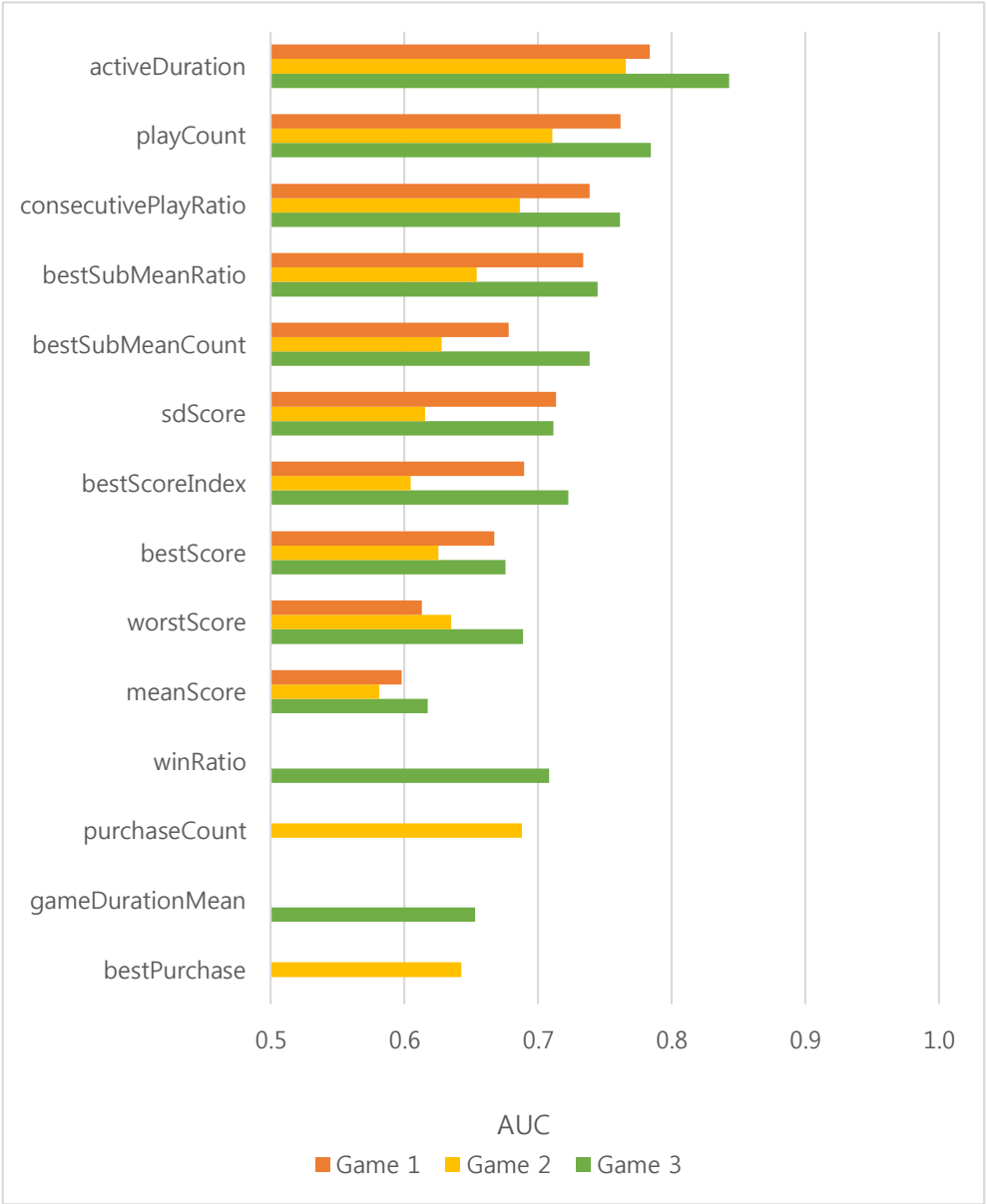

그림 70 Gradient boosting 결과

세개의 게임 중에서 Game 3이 가장 높은 예측 성능을 보여주고

Game 2가 가장 낮은 예측 성능을 보여주고 있다. 또한, 세개의 게임 모두다 activeDuration, playCount를 가지고 예측을 했을 때가 예측 성능이 가장 높게 나왔다. 세개의 게임 전체에서 AUC가 0.7이상인 Feautre는 15개이다. Game 2의 전용 Feature인 purchaseCount도 0.7이 약간 안되는 성능을 보여주고 있다. 또한, Game 3의 전용 Feature인 winRatio는 0.7이상의 성능을 보여주고 있다. meanScore는 세게임 모두 다 가장 낮은 성능을 보여주고 있다.

Single feature AUC: Logistic regression

아래의 [그림 71]은 Single feature를 Logistic regression에 넣고 돌린 결과이다.

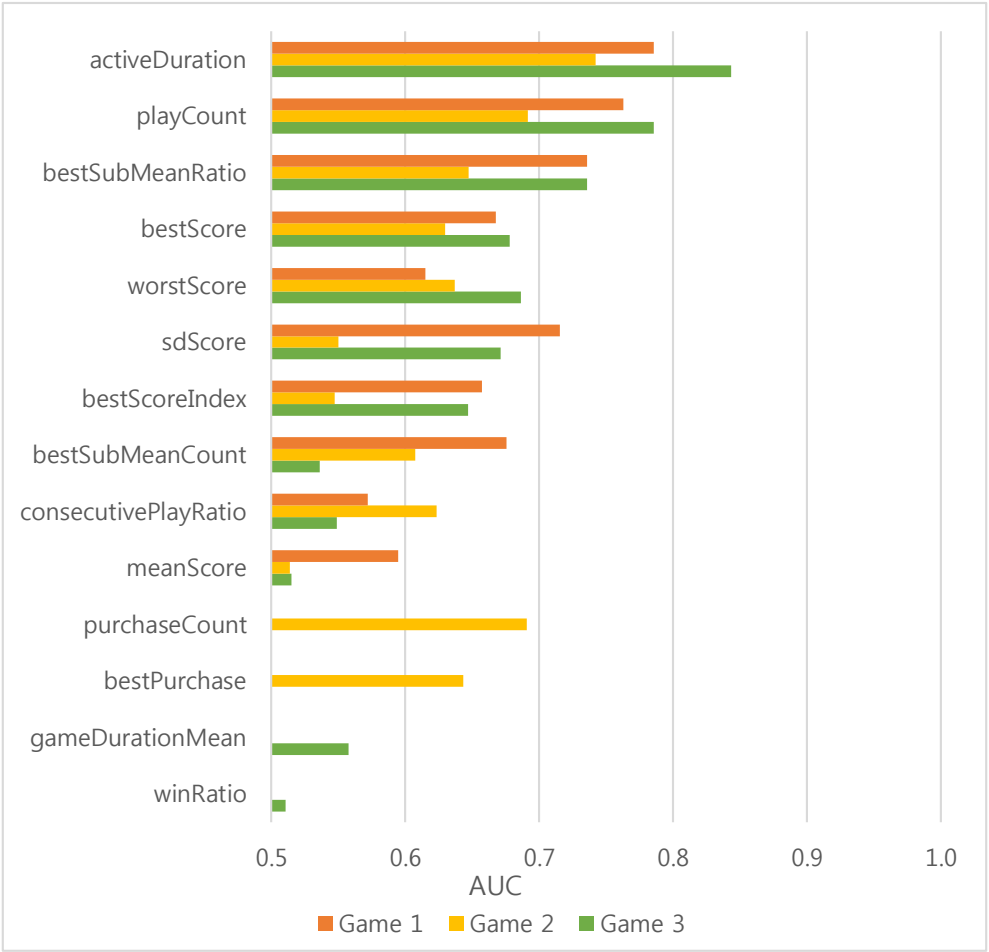

그림 71 Logistic regression 결과

Gradient boosting와 비슷하게도 Game 3가 세개의 게임 중에 가장 높은 성능을 보여주고 있고 Game 2가 가장 낮은 성능을 보여주고 있다. 전체적으로는 Gradient boosting보다 낮은 성능을 보여주고 있다. 세개의 게임을 합쳐 AUC가 0.7이상인 Feature는 8개밖에 되지 않는다. Feature의 상위 성능 순위 중 activeDuration, playCount순서는 Gradient boosting와 같다. Game 2의 전용 Feature인 purchaseCount는 Gradient boosting 때와 비슷한 값으로 나온다.

Single feature AUC: Random forest

아래의 [그림 72]은 Single feature를 Random forest에 넣고 돌린 결과이다

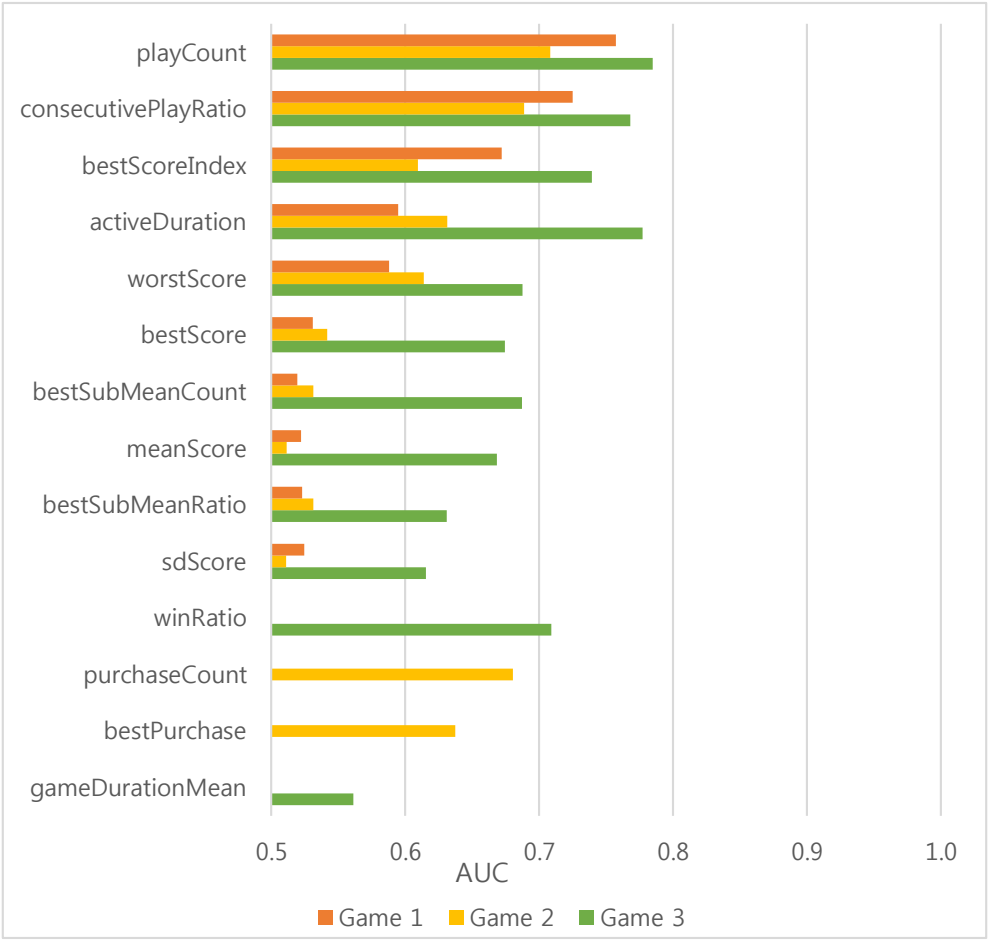

그림 72 Random forest 결과

Random forest에서는 activeDuration값의 Game 1, Game 2값이 0.7 이하로 떨어졌다. 또한, consecutivePlayRatio 값이 상위 성능으로 올라왔다. 전체적으로 Gradient boosting보다 낮은 성능을 보여주고 있다. 세개의 게임 전체 Feature 중 0.7이상이 되는 Feature는 총 8개로 Gradient boosting의 15개에 비해 낮은 수치를 기록하고 있다.

Feature selection

아래의 [표 18]은 Single feature의 Feature selection의 순위이다.

Selection order	Game 1	Game 2	Game 3
1	activeDuration	activeDuration	activeDuration
2	bestScore	playCount	worstScore
3	playCount	bestPurchase	winRatio
4	bestScoreIndex	consecutivePlayRatio	playCount
5	sdScore	worstScore	bestScoreIndex
6	consecutivePlayRatio	bestScoreIndex	bestScore
7	worstScore	bestSubMeanCount	bestSubMeanCount
8	meanScore	sdScore	bestSubMeanRatio
9	bestSubMeanRatio	meanScore	sdScore
10	bestSubMeanCount	bestScore	consecutivePlayRatio
11	-	purchaseCount	meanScore
12	-	bestSubMeanRatio	gameDurationMean

표 18 Feature selection 결과

위의 표와같이 첫번째 Feature는 activeDuration으로 동일하다. 그 이후로는 세개의 게임의 순위가 다르다. Game 2에서의 전용 Feature인 bestPurchase가 3번째이고, Game 3에서의 전용 Feature인 winRatio도 3번째이다. 또한, playCount도 Game 1에서 3번째, Game 2에서 2번째, Game 3에서 4번째로 높은 순위를 기록하고

있다.

Feature importance

아래의 [그림 73]은 Single feature의 Feature importace를 추출한 결과이다.

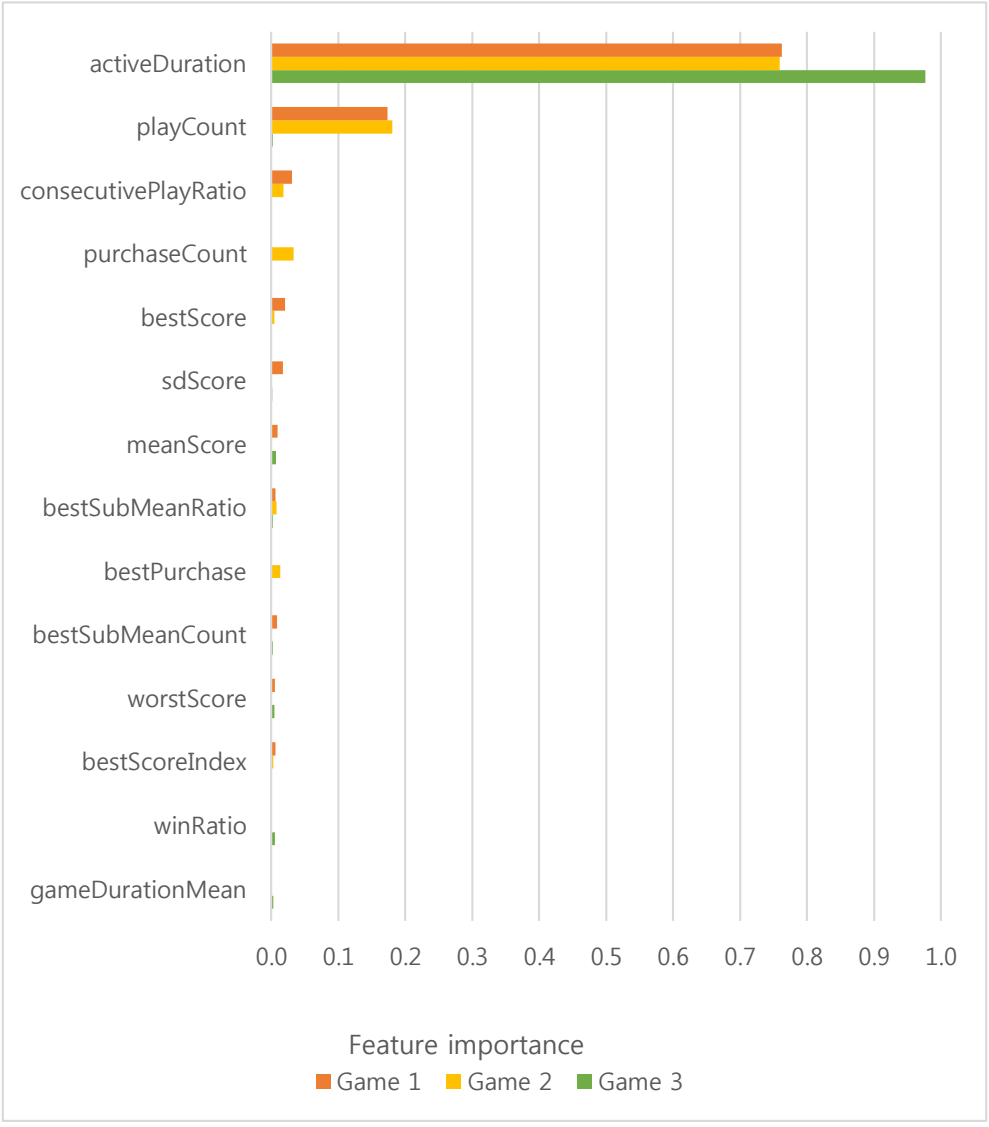

그림 73 Feature importance 결과

그림에서와 같이 activeDuration이 다른 Feature들에 비해 월등히 높게 나온다. Game 3은 activeDuration하나만으로 1에 육박할

만큼 좋게 나왔다. 또한, Game 1, Game 2는 playCount까지만 높은 importance를 가지고 있다.

Overall ranking

아래 세개의 표는 각 게임마다 Feature 별 랭킹과 평균랭킹을 구하였다[표 19, 표 20, 표 21]. 또한, [표 22]에 세개 게임의 카테고리별 평균 랭킹을 종합하였다.

	Rank Avg.	correlation		gradient boosting		random forest		logistic regression		forward selection		importance	
		rank	value	rank	value	rank	value	rank	value	rank	value	rank	value
active Duration	1	1	−0.380	1	0.7834	4	0.5948	1	0.7857	1	0.7834	1	0.7626
playCount	2	2	−0.265	2	0.7616	1	0.7573	2	0.7628	3	0.7955	2	0.1736
bestScore	3	5	−0.103	8	0.6672	6	0.5308	6	0.6674	2	0.7901	4	0.0200
consecutive PlayRatio		7	−0.095	3	0.7385	2	0.7249	10	0.5720	6	0.7960	3	0.0311
sdScore	5	6	−0.097	5	0.7133	7	0.5246	4	0.7155	5	0.7955	5	0.0173
bestScore Index	6	4	0.149	6	0.6893	3	0.6718	7	0.6571	4	0.7961	9	0.0061
bestSub MeanRatio	7	3	−0.256	4	0.7336	8	0.5229	3	0.7356	9	0.7934	8	0.0064
bestSub MeanCount	8	8	−0.053	7	0.6778	10	0.5193	5	0.6756	10	0.7924	7	0.0085
worstScore	9	9	0.037	9	0.6130	5	0.5878	8	0.6149	7	0.7956	10	0.0051
meanScore	10	10	−0.018	10	0.5979	9	0.5223	9	0.5945	8	0.7946	6	0.0088

표 19 Game 1 Overall single features ranking

	Rank Avg.	correlation		gradient boosting		random forest		logistic regression		forward selection		importance	
		rank	value	rank	value	rank	value	rank	value	rank	value	rank	value
Active Duration	1	1	−0.3391	1	0.7657	5	0.6313	1	0.7422	1	0.7657	1	0.7594
playCount	2	2	−0.2857	2	0.7106	1	0.7084	2	0.6916	2	0.7657	2	0.1802
Purchase Count	3	3	−0.2135	3	0.6879	3	0.6804	3	0.6907	11	0.7700	3	0.0328

Best Purchase	4	5	−0.0664	6	0.6426	4	0.6373	5	0.6433	3	0.7706	5	0.0131
consecutive PlayRatio		6	0.0628	4	0.6864	2	0.6887	8	0.6234	4	0.7724	4	0.0179
bestSub MeanRatio	6	4	−0.1310	5	0.6538	10	0.5312	4	0.6474	12	0.76963	6	0.0073
worstScore	7	11	−0.0011	7	0.6350	6	0.6138	6	0.6370	5	0.7724	9	0.0001
bestScore	8	8	0.0018	9	0.6255	8	0.5416	7	0.6298	10	0.7709	7	0.0042
bestScore Index	9	7	0.0384	11	0.6048	7	0.6092	11	0.5472	6	0.7724	8	0.0028
bestSubMeanCount	10	10	0.0013	8	0.6278	9	0.5315	9	0.6074	7	0.7723	9	0.0001
sdScore	11	9	0.0017	10	0.6155	12	0.5110	10	0.5500	8	0.7720	9	0.0001
meanScore	12	12	−0.0009	12	0.5809	11	0.5113	12	0.5136	9	0.7718	9	0.0001

표 20 Game 2 Overall single features rank

	Rank Avg.	correlation		gradient boosting		random forest		logistic regression		forward selection		importance	
		rank	value	rank	value	rank	value	rank	value	rank	value	rank	value
Active Duration	1	1	−0.636	1	0.8428	2	0.7773	1	0.8434	1	0.8428	1	0.9770
playCount	2	3	−0.244	2	0.7841	1	0.7848	2	0.7855	4	0.8500	6	0.0022
worstScore	3	4	0.239	9	0.6888	6	0.6873	4	0.6863	2	0.8471	4	0.0046
bestScore Index	4	6	0.189	6	0.7224	4	0.7394	7	0.6469	5	0.8500	10	0.0001
winRatio	5	8	−0.091	8	0.7083	5	0.7088	12	0.5106	3	0.8495	3	0.0052
bestSub MeanRatio	6	7	−0.113	4	0.7445	10	0.6308	3	0.7355	8	0.8499	8	0.0019
bestScore	7	2	−0.252	10	0.6756	8	0.6742	5	0.6778	6	0.8500	10	0.0001
consecutive PlayRatio	8	10	−0.030	3	0.7613	3	0.7681	9	0.5488	10	0.8497	10	0.0001
sdScore	10	5	−0.208	7	0.7113	11	0.6153	6	0.6711	9	0.8499	0	0.0016
bestSub MeanCount		11	0.0269	5	0.7384	7	0.6871	10	0.5360	7	0.8500	7	0.0021

game DurationMean	11	9	−0.035	11	0.6527	12	0.5610	8	0.5575	12	0.8480	5	0.0030
meanScore		12	−0.005	12	0.6173	9	0.6683	11	0.5151	11	0.8492	2	0.0070

표 21 Game 3 Overall single features rank

순위	Game_1	Game_2	Game_3
1	activeDuration	activeDuration	activeDuration
2	playCount	playCount	playCount
3	bestScore consecutivePlayRatio	purchaseCount	worstScore
4		bestPurchase consecutivePlayRatio	bestScoreIndex
5	sdScore		winRatio
6	bestScoreIndex	bestSubMeanRatio	bestSubMeanRatio
7	bestSubMeanRatio	worstScore	bestScore
8	bestSubMeanCount	bestScore	consecutivePlayRatio sdScore
9	worstScore	bestScoreIndex	
10	meanScore	bestSubMeanCount	bestSubMeanCount gameDurationMean
11	-	sdScore	
12	-	meanScore	meanScore

표 22 Overall single features average rank

세개의 게임 모두 activeDuration, playCount가 상위권에 랭크되어있다. 또한, 특정 게임에만 랭크가 높은 Feature들이 있다. bestScore, consecutivePlayRatio, worstScore, bestScoreIndex feature가 특정 게임에만 랭크가 높다. 또한, 전용 Feature 중에 purchaseCounat와 bestPurchase가 높은 랭크를 차지하고 있다. meanScore는 세개의 게임 모두 제일 낮은 랭크를 차지하고 있다.

2. 이탈 예측 모델 실험 결과: 관찰 기간 및 이탈예측 기간 별

이탈 예측 모델에 대한 실험 결과를 관찰 기간과 이탈예측 기간을 변경함에 따라서 예측 성능에 어떤 영향을 끼치는지 알아보기로

한다. 예측에서 사용될 데이터 테이블들의 값은 관찰 기간과 이탈예측 기간이 변화함에 따라서 값들이 변화한다. 관찰 기간이 변화하면 데이터 테이블의 유저 별 Feature 값들이 변화한다. 또한, 이탈예측 기간을 변화하면 유저 별 이탈 여부가 변화한다. 이러한 변화가 이탈 예측 모델의 ROC curve의 AUC 결과값을 변화시킨다. 본 실험 결과는 세개의 알고리즘 중 성능이 제일 좋은 Gradient boosting으로 실험을 하였으며 나머지 알고리즘에 대한 실험 결과는 알고리즘 별 이탈 예측 모델 실험 결과에서 확인 할 수 있다. 또한, 사용된 Feature는 Forward selection을 통해서 가장 성능이 좋은 Feature 조합을 사용하였다.

Game 1

다음은 Game 1의 관찰기간 및 이탈예측 기간 별 이탈 예측 모델의 AUC이다[그림 74]. 가로축은 관찰 기간(Observastion period, OP), 깊이축은(Churn prediction period, CP), 높이축은 ROC curve의 AUC값이다.

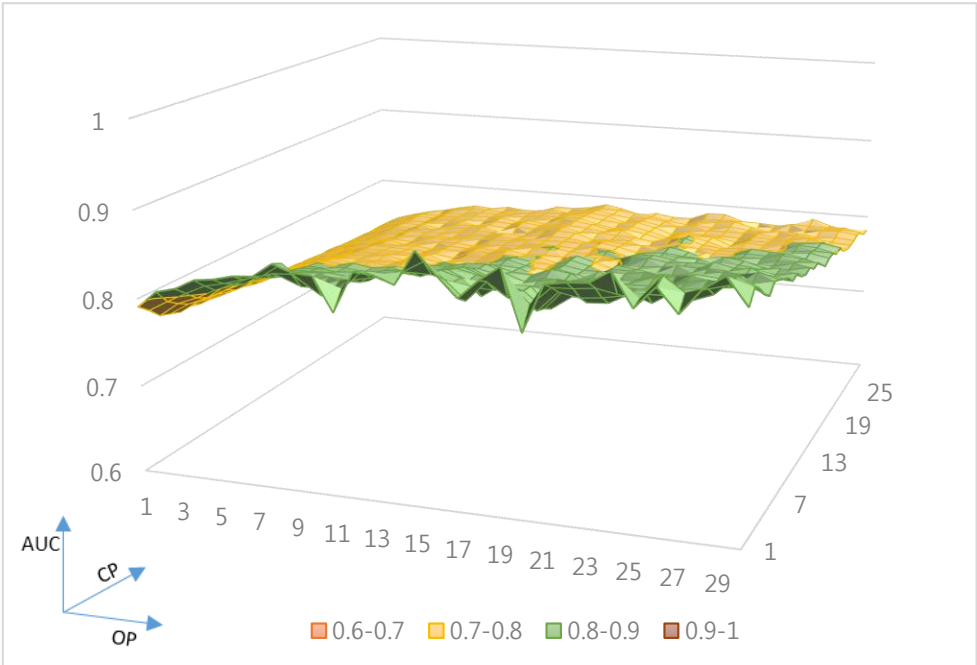

그림 74 Game 1 결과

관찰 기간이 길수록, 이탈예측 기간이 짧을수록 예측 성능이

좋아지고 있다. 또한, 이탈예측 기간이 하루였을 때는 관찰기간이 길어짐에 따라 예측 성능의 편차가 크고 변화량도 크게 나타난다. 그리고 관찰 기간과 유저예측 기간이 게임 초반 기간을 넘은 다음엔 AUC의 큰 변화가 없다. 다음 [그림 75]은 위의 Game 1의 결과 중 4개의 관찰 기간을 선정하여 이탈예측 기간 별 AUC의 변화 그래프이다.

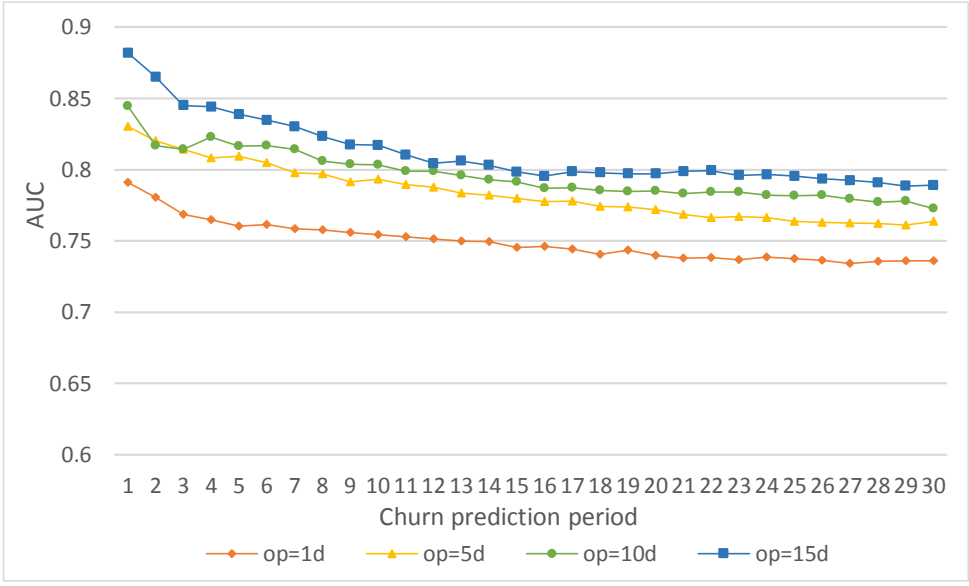

그림 75 Game 1의 이탈 예측 기간 별 AUC

앞에서 설명한 바와 같이 이탈예측 기간이 길어짐에 따라서 예측 성능이 줄어든다. 또한, 관찰 기간이 길어짐에 따라서 예측 성능이 높아진다. 관찰 기간의 증가에 따른 AUC값의 증가폭도 볼 수가 있는데, 관찰 기간이 1일에서 5일로 증가됐을 때의 AUC의 증가폭이 10일에서 15일로 증가됐을 때 폭보다 크게 나타났다.

Game 2

다음은 Game 2의 관찰기간 및 이탈예측 기간 별 이탈 예측 모델의 AUC이다[그림 76]

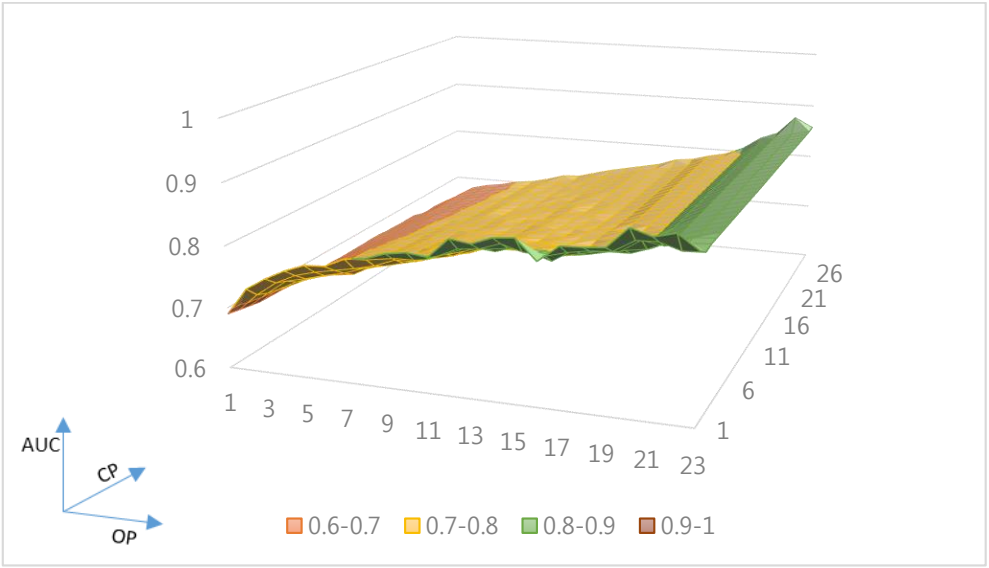

그림 76 Game 2 결과

전체 적으로 관찰 기간이 증가함에 따라서 상승하는 그래프이다. 또한, 이탈 예측기간이 짧을수록 예측성능이 커진다. 초반 기간 전까지의 이탈예측 기간에서 AUC의 변화량는 초반기간 이후의 변화량 대비 많은 변화량을 보여준다. 그리고 관찰 기간이 증가함에 따라 AUC가 선형적으로 상승하는 것을 볼 수 있다. 다음 [그림 77]은 위의 Game 2의 결과 중 4 개의 관찰 기간을 선정하여 이탈예측 기간 별 AUC의 변화 그래프이다.

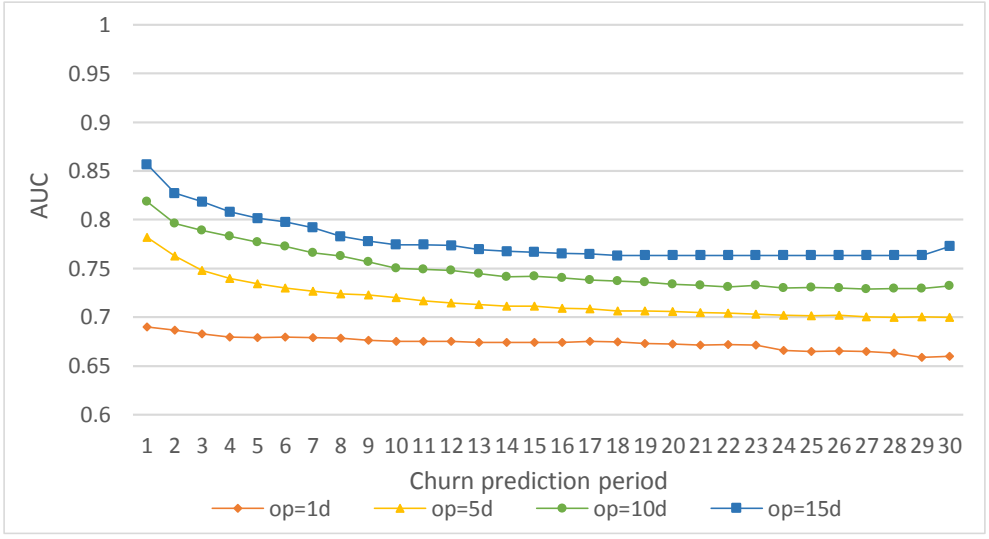

그림 77 Game 2의 이탈예측 기간 별 AUC

Game 1의 그래프와 비슷하게 관찰 기간이 길어질수록, 유저이탈 예측기간이 짧을수록 예측성능이 좋게 나온다. 그러나 관찰 기간이 커짐에 따라 AUC의 증가폭은 Game 1과는 다르게 나타났다. Game 1의 경우는 점점 증가폭이 줄어들지만 Game 2는 선형적으로 증가가된다. 또한, 이탈 예측 기간이 10일부터 30일까지는 AUC의 변화없는 안정적인 예측 성능을 보여주고 있다.

Game 3

다음은 Game 3의 관찰기간 및 이탈예측 기간 별 이탈 예측 모델의 AUC이다[그림 78]

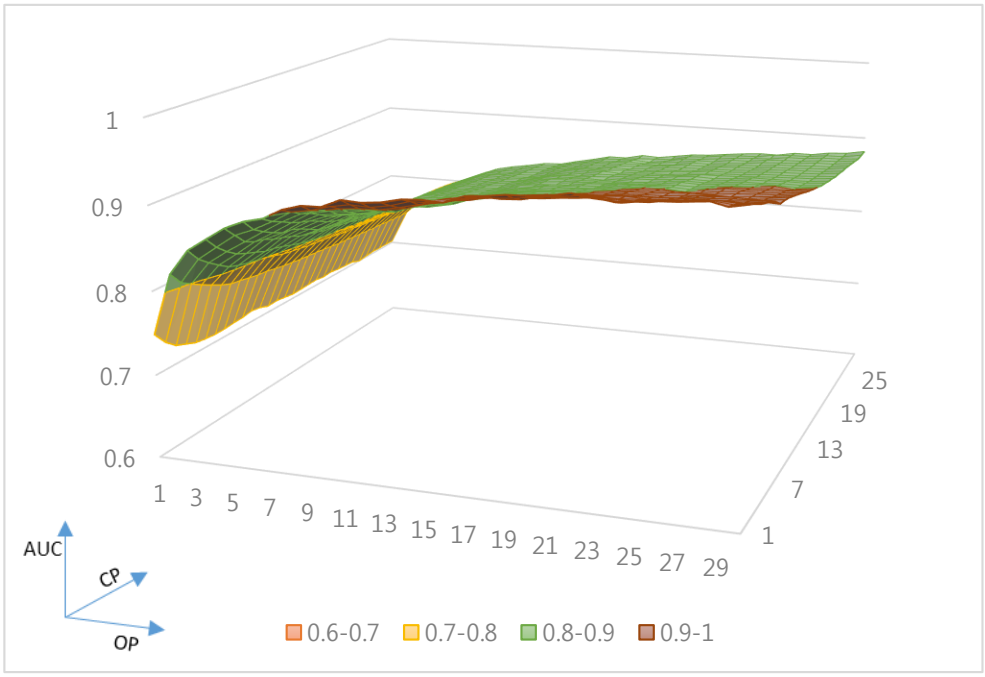

그림 78 Game 3 결과

전체적으로 관찰 기간이 길어짐에 따라서 예측 성능이 수렴하는 형태로 증가하고 있다. 또한, 이탈예측 기간이 짧을수록 예측 성능이 높다. 초반 기간까지의 관찰기간에서 증가되는 AUC값이 초반기간 이후의 증가되는 값보다 높게 증가가된다. 또한, 관찰 기간과 이탈예측 기간이 초반기간을 경과한 후의 AUC의 변화는 눈에 띄게 줄어든다.

다음 [그림 79]은 위의 Game 2의 결과 중 4 개의 관찰 기간을 선정하여 이탈예측 기간 별 AUC의 변화 그래프이다.

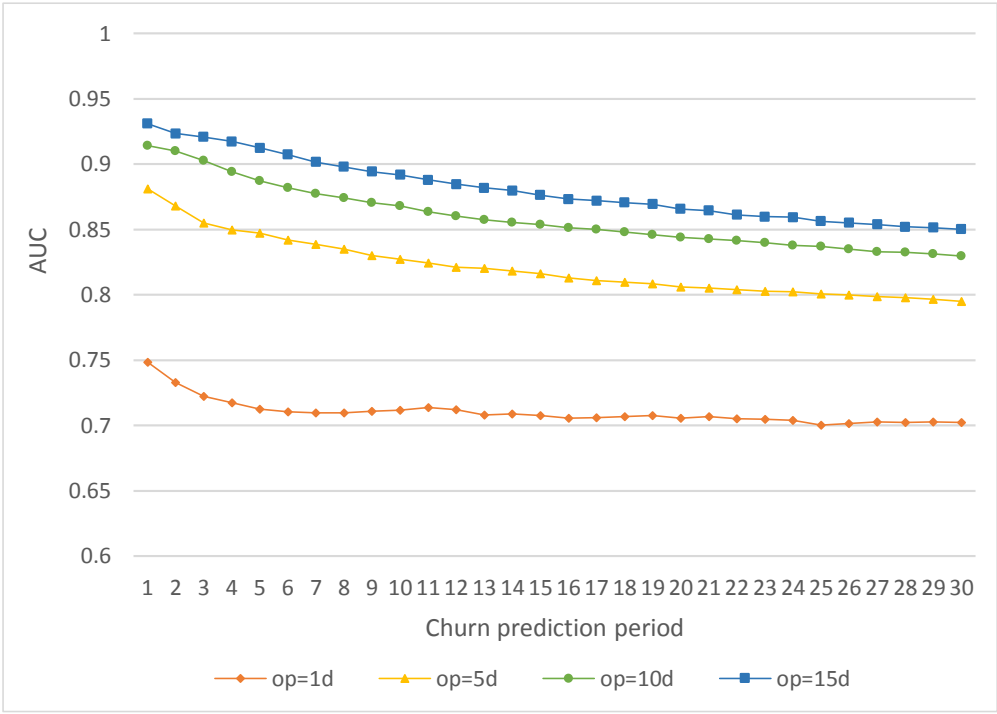

그림 79 Game 3의 이탈 예측 기간 별 AUC

Game 1과 비슷하게 관찰 기간이 길어짐에 따라 증가되는 AUC의 폭이 줄어든다. 관찰 기간이 1일에서 5일로 길어지면서 증가되는 AUC의 양이 10일에서 15일로 길어지면서 증가되는 양보다 훨씬 많다. 이탈예측 기간이 초반기간 이후부터 AUC가 큰 변화 없이 안정적예측 성능을 보여준다.

게임 별 비교

[그림 80]은 관찰 기간과 이탈예측 기간의 비율을 1:1로 유지하면서 기간 별 세 게임의 AUC의 변화를 나타낸 그래프이다.

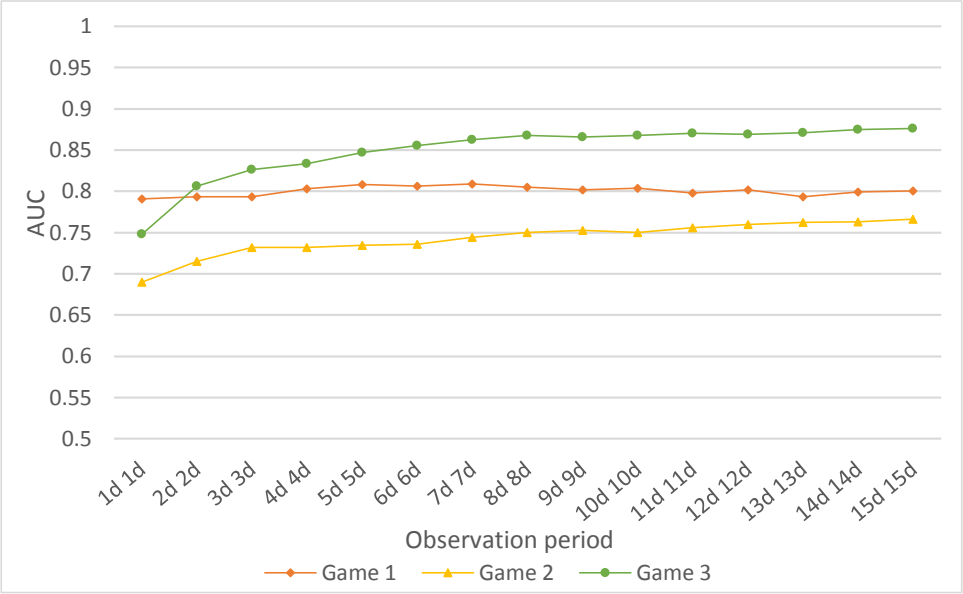

그림 80 세 게임 별 관찰기간 및 이탈예측 기간의 변화에 따른 AUC

예측 성능은 Game 3, Gam1, Game 2순으로 높게 나타났다. Game 1의 그래프는 관찰 기간과 이탈예측 기간의 변화함에도 AUC값은 큰 변화가 없다. Game 2의 그래프는 두 기간이 변화함에 따라서 AUC값은 증가하지만 증가 폭이 Game 3에 비해 적다. Game 3은 두 기간이 1일 일때는 AUC의 값이 Game 1보다 작았지만 2일 째부터 Game 1보다 높게 나왔다. 또한, 초반 기간까지 증가를 하다가 초반 기간 이후부터는 AUC의 큰 변화가 없는 수렴하는 형태로 바뀌었다. 이렇게 Game 3이 관찰기간 초반에 증가하는 양이 많은 이유는 첫날엔 유저들의 플레이가 적어서 유저들에 대한 정보가 적어서 예측 성능도 좋지 않다고 해석이 가능하다.

3. 이탈 예측 모델 실험 결과: 알고리즘 별

Gradient boosting, Logistic regression, Random forest 알고리즘 별로 이탈 예측의 성능은 어떠한지 알아보기로 한다. 세개의 알고리즘 별로 만들어진 각 게임 별 900개, 총 2,700개의 데이터 테이블을 가지고 유저 이탈 모델을 만들고 ROC curve의 AUC를 살펴본다. 유저 이탈 예측 모델 생성시 사용된 Feature는 Forward selection을 통해 가장 성능이 높은 Feature들의 조합으로 유저 이탈

예측 모델을 생성하였다.

Game 1

다음은 Game 1의 3개의 알고리즘에 대한 관찰기간 및 이탈예측 기간 별 이탈 예측 모델의 AUC이다 [그림 81, 그림 82, 그림 83]. 가로축은 관찰 기간(Observastion period, OP), 깊이축은(Churn prediction period, CP), 높이축은 ROC curve의 AUC값이다.

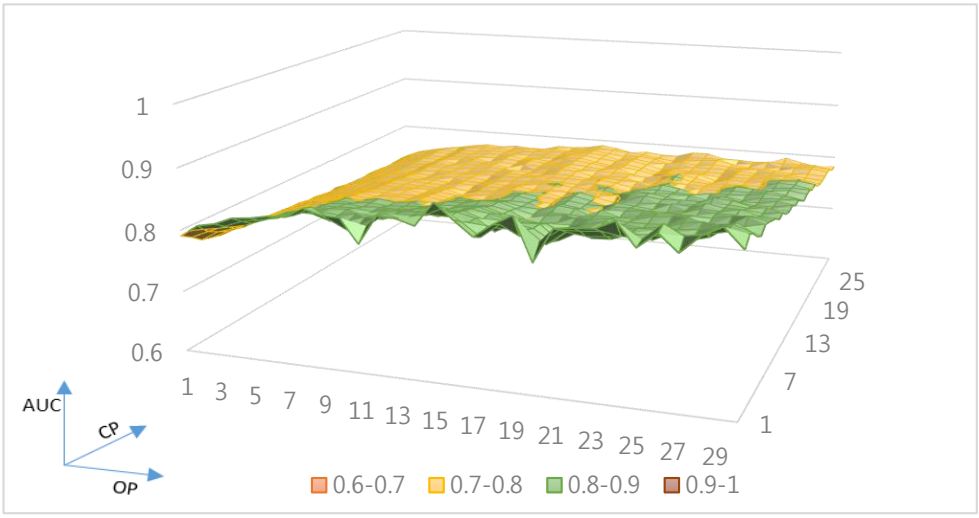

그림 81 Game 1 Gradient boosting 결과

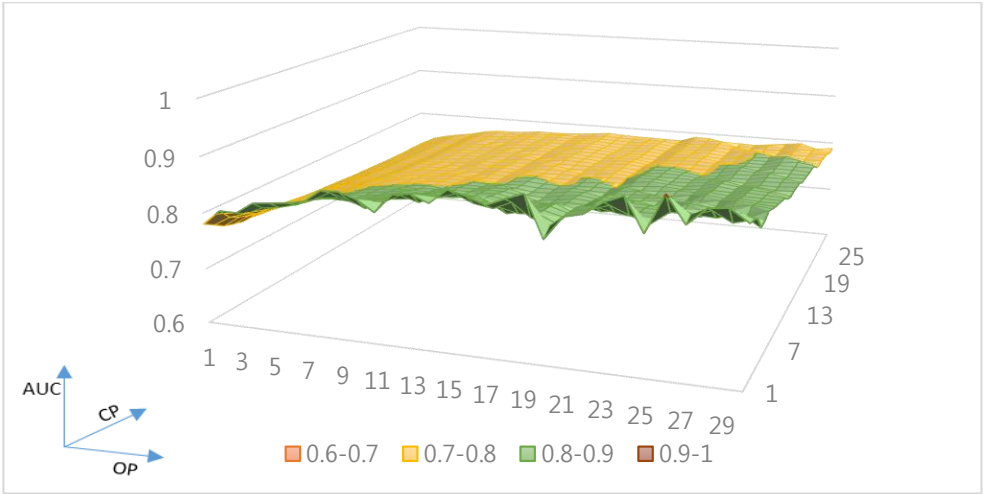

그림 82 Game 1 Logistic regression 결과

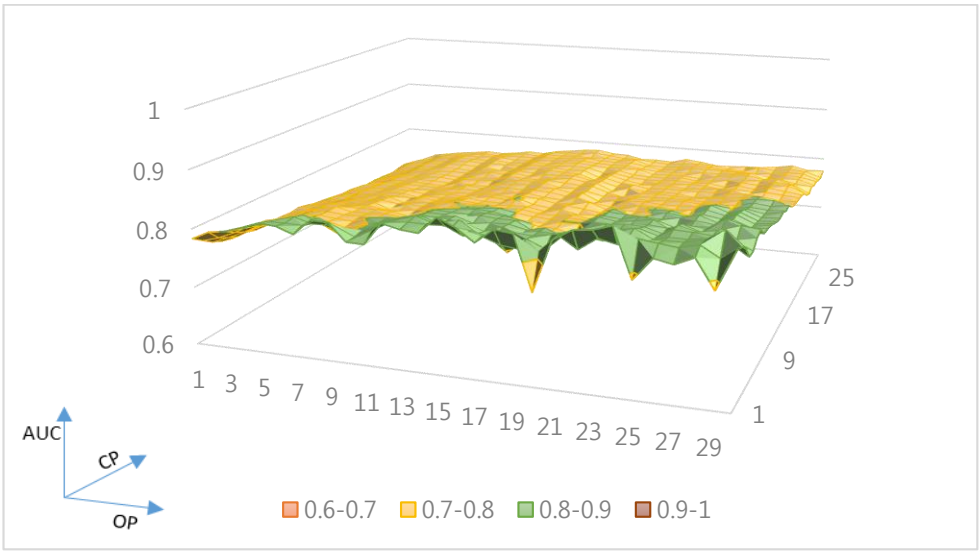

그림 83 Game 1 Random forest 결과

아래 세 그래프들은 Game 1의 각 알고리즘 별 구체적인 차이를 나타는 그래프들이다. [그림 84]의 AUC는 Gradient boosting 모델의 AUC에 Logistic regression 모델의 AUC을 뺀 수치, [그림 85]의 AUC는 Gradient boosting 모델의 AUC에 Random forest 모델의 AUC을 뺀 수치, [그림 86]은 Logistic regression 모델에 Random forest모델을 뺀 수치이다.

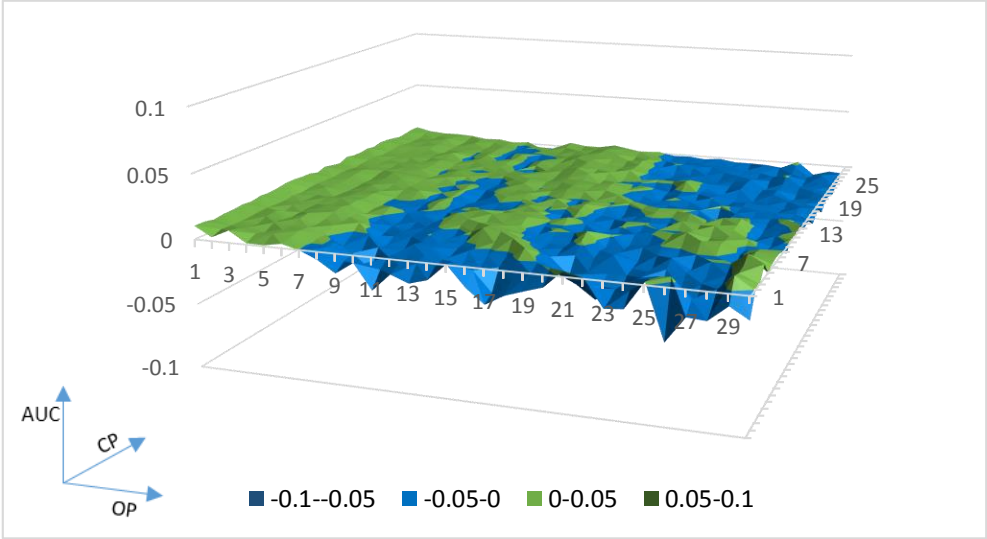

그림 84 Game 1 Gradient boosting과 Logistic regression의 AUC 차이

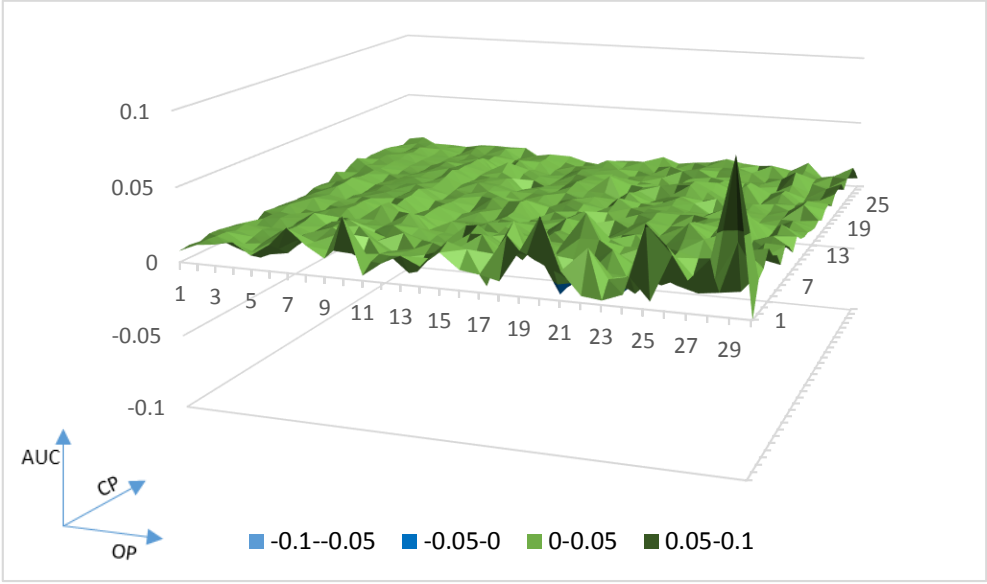

그림 85 Game 1 Gradient boosting과 Random forest의 AUC 차이

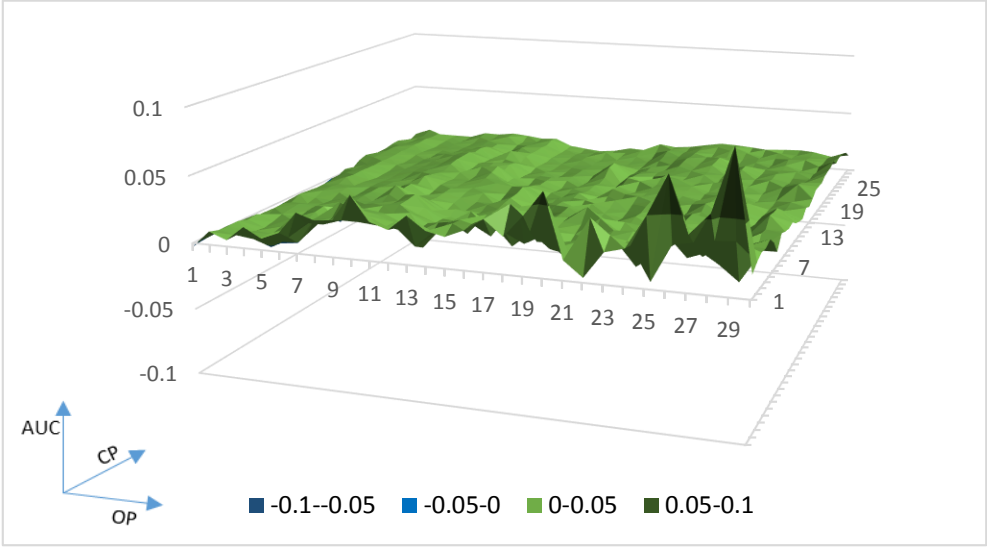

그림 86 Game 1 Logistic regression과 Random forest의 AUC 차이

위에서 Game 1의 세개의 알고리즘 중 두개씩 짝을 지어 AUC값 차이를 비교 해보았다. 세개 알고리즘에 대한 차이를 한번에 보기 위하여, 세개의 알고리즘에 대한 표준 편차 그래프를 다음과 같이 그려보았다[그림 87].

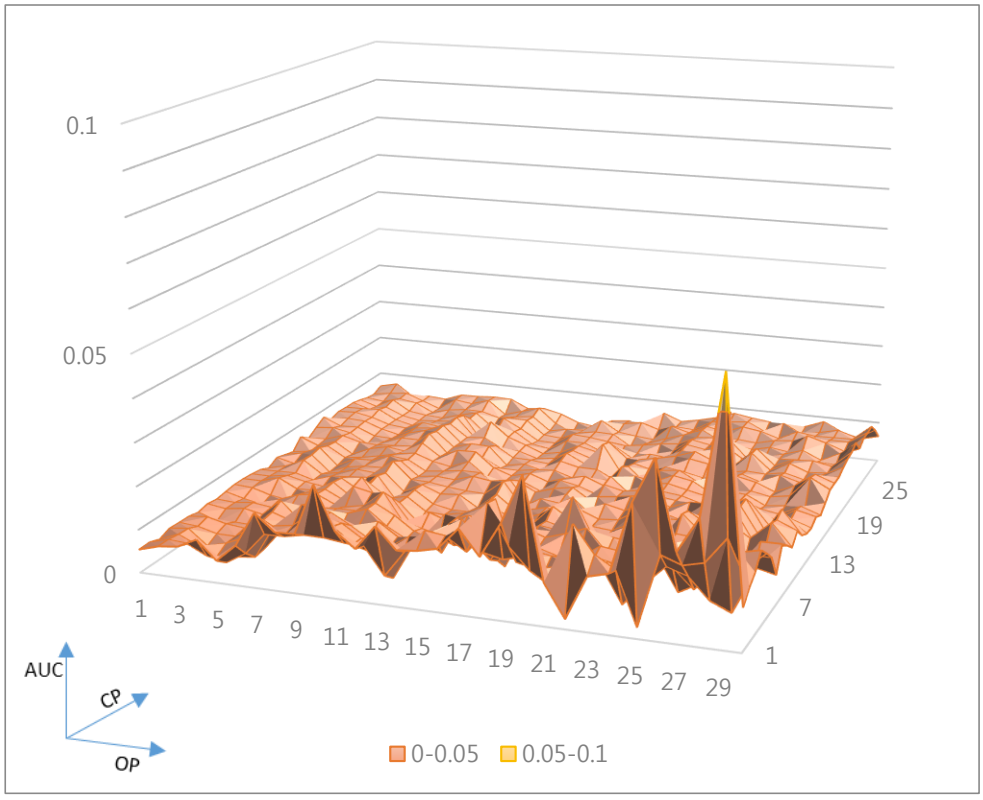

그림 87 Game 1 세 알고리즘의 표준편차

위와같이 3개의 알고리즘으로 생성된 이탈 예측모델은 서로 비슷한 값 형태를 나타내고있다. 세개의 알고리즘 중 관찰 기간이 20일 이하인 AUC값들은 Gradient boosting의 성능이 가장 높았으며 20일 이후의 AUC값들은 Logistic regression의 성능이 가장 높았다. 그러나 표준 편차의 크기를 살펴보면 가장 큰 값은 0.06이고, 대부분은 0.01 부근에서 분포되어있다. 이는 실제로 Game 1에 대한 알고리즘 별로 예측 성능의 차이는 아주 작다고 해석이 가능하다.

Game 2

다음은 Game 2의 3개의 알고리즘에 대한 관찰기간 및 이탈예측 기간 별 이탈 예측 모델의 AUC이다 [그림 88, 그림 89, 그림 90].

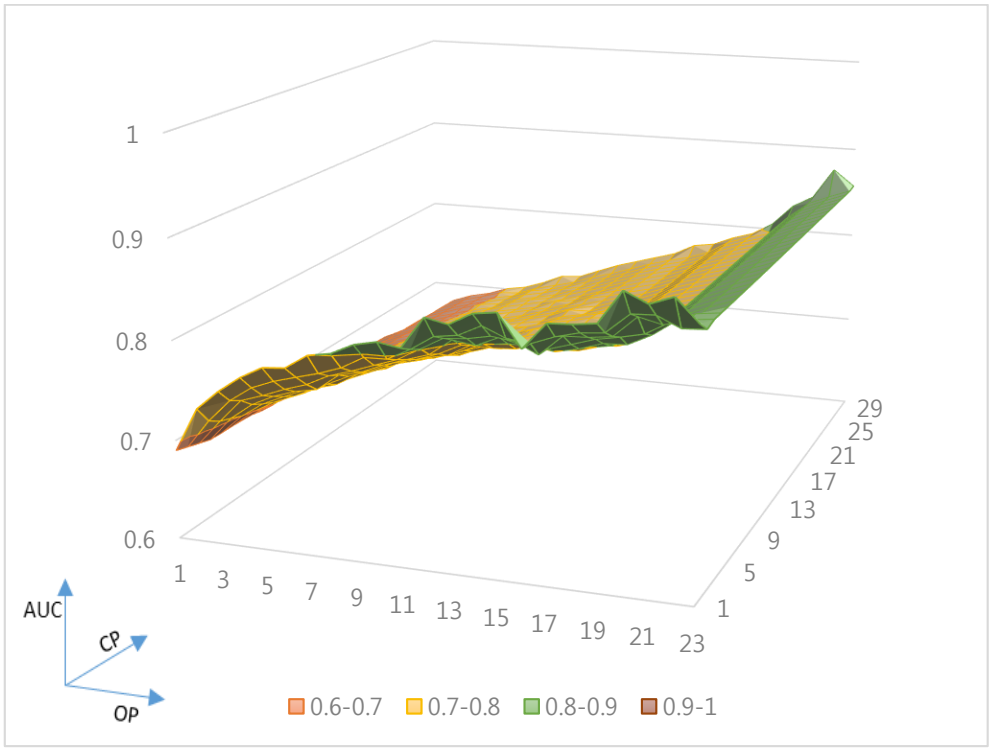

그림 88 Game 2 Gradient boosting 결과

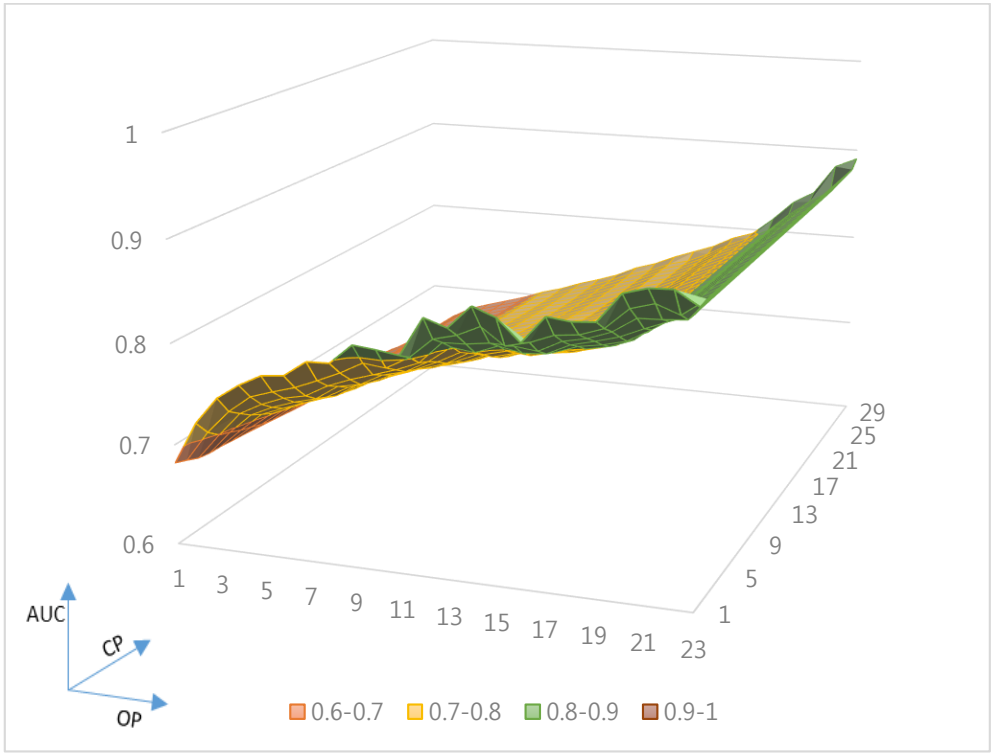

그림 89 Game 2 Logistic regression 결과

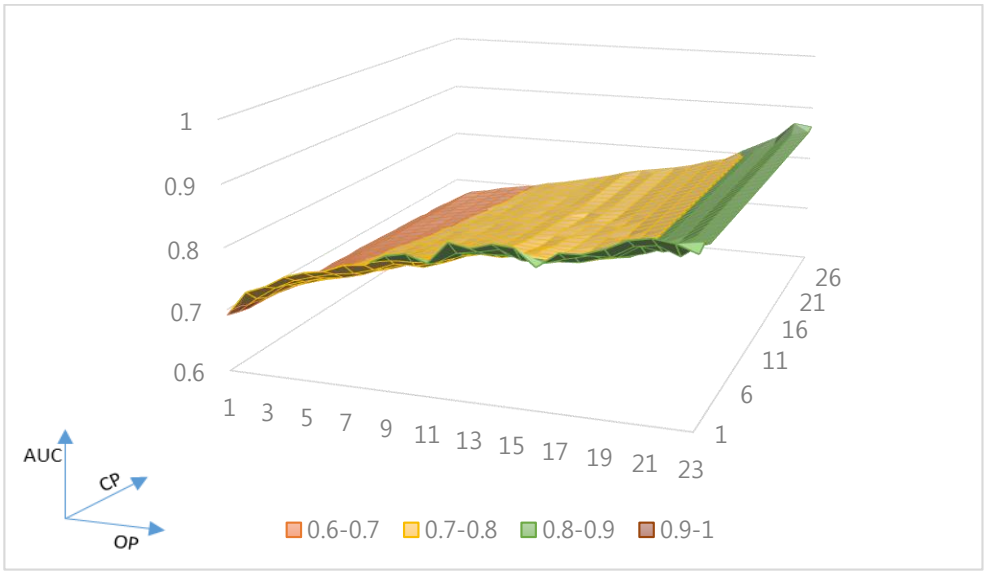

그림 90 Game 2 Random forest 결과

아래 세 그래프들은 Game 2의 각 알고리즘 별 구체적인 차이를 나타는 그래프들이다. [그림 91]의 AUC는 Gradient boosting 모델의 AUC에 Logistic regression 모델의 AUC을 뺀 수치, [그림 92]의 AUC는 Gradient boosting 모델의 AUC에 Random forest 모델의 AUC을 뺀 수치, [그림 93]은 Logistic regression 모델에 Random forest모델을 뺀 수치이다.

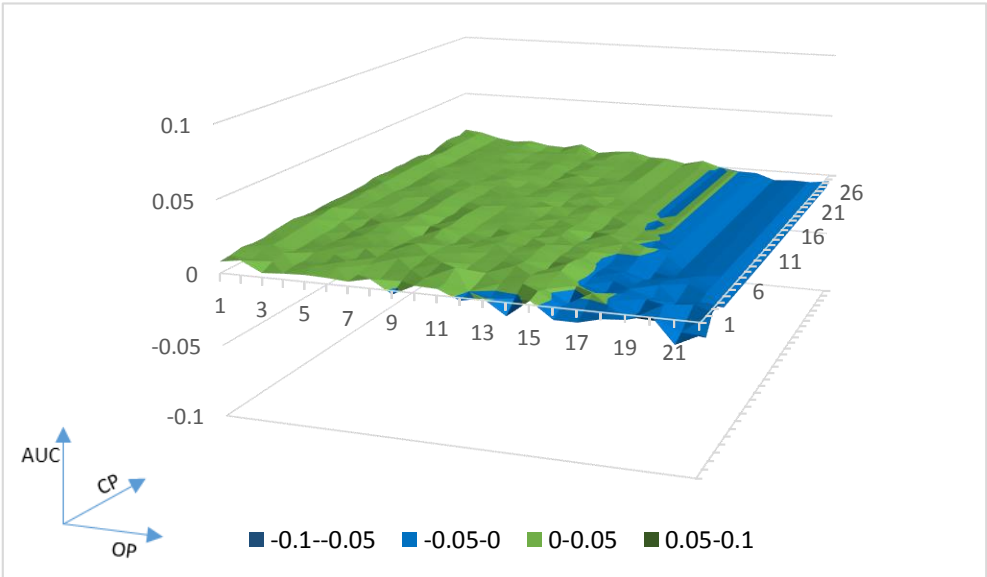

그림 91 Game 2 Gradient boosting과 Logistic regression의 AUC 차이

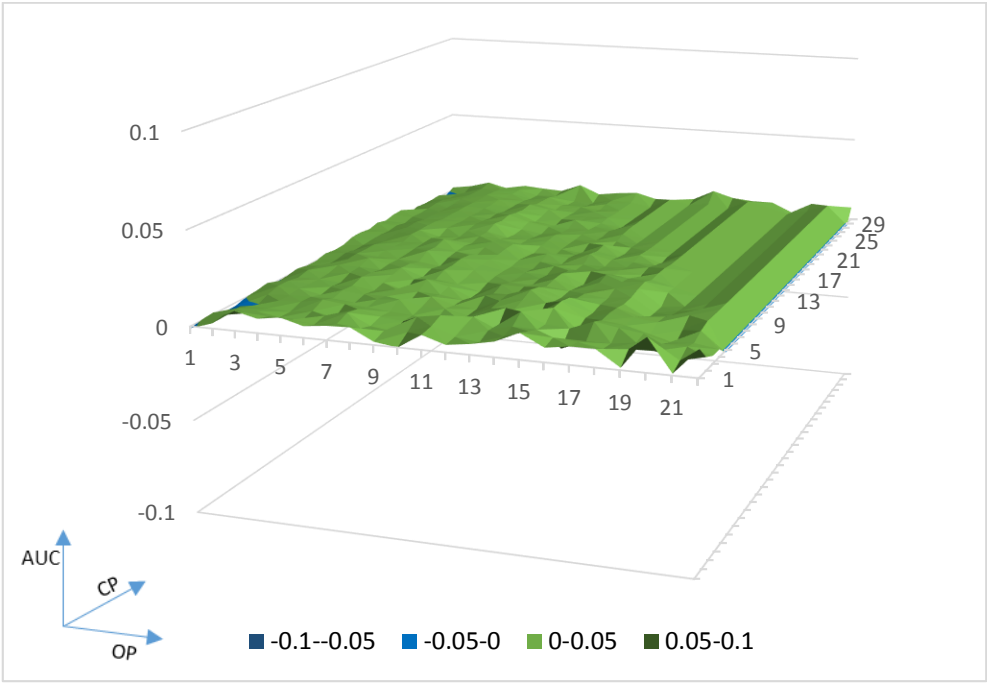

그림 92 Game 2 Gradient boosting과 Random forest의 AUC 차이

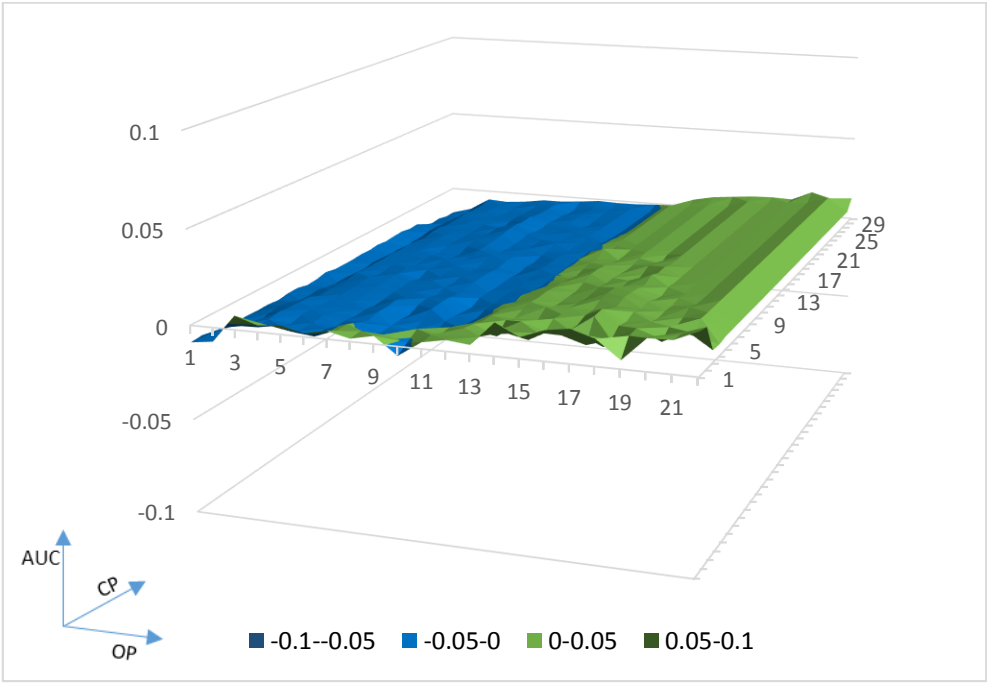

그림 93 Game 2 Logistic regression과 Random forest의 AUC 차이

위에서 Game 2의 세개의 알고리즘 중 두개씩 짝을 지어 AUC값 차이를 비교 해보았다. 세개 알고리즘에 대한 차이를 한번에

보기 위하여, 세개의 알고리즘에 대한 표준 편차 그래프를 다음과 같이 그려보았다[그림 94].

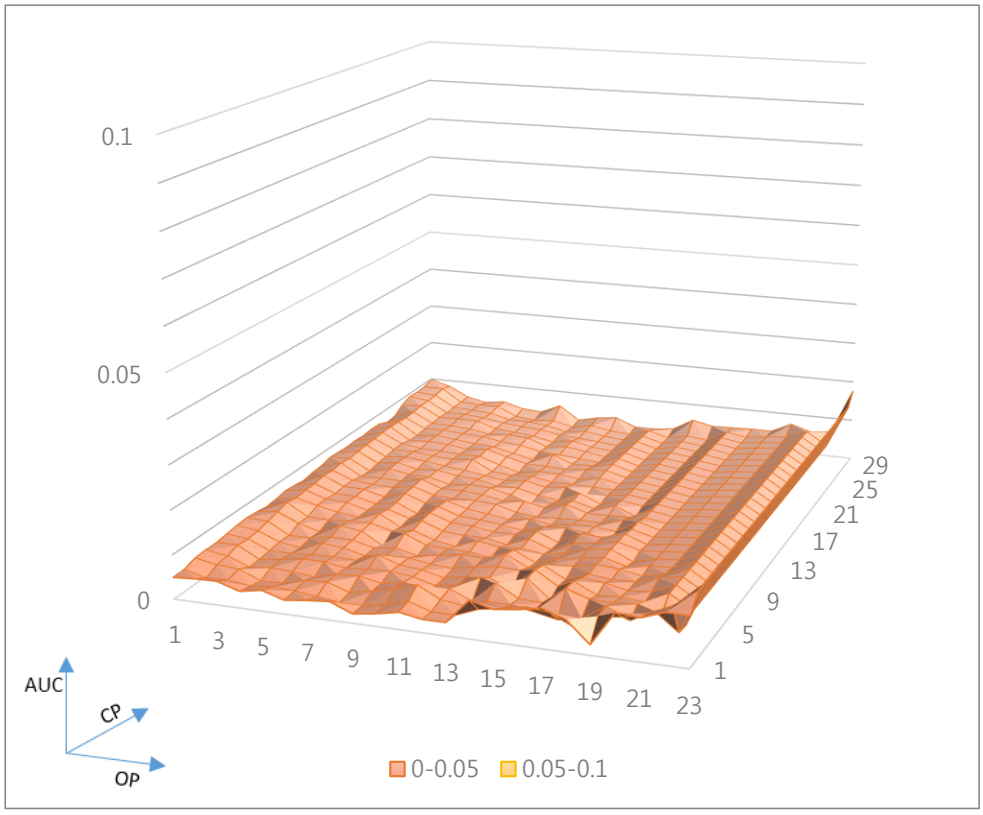

그림 94 Game 2 세 알고리즘의 표준편차

Game 1과 비슷하게, 세개의 알고리즘에 대한 그래프의 형태도 비슷하게 나왔다. 또한, 알고리즘 중 관찰 기간이 16일 이하인 AUC값들은 Gradient boosting의 성능이 가장 높았으며 16일 이후의 AUC값들은 Logistic regression의 성능이 가장 높았다. 그러나 전체 게임의 표준 편차의 크기를 살펴보면 가장 큰 값은 약 0.02이고, 대부분은 0.01 부근에서 분포되어있다. 이는 실제로 Game 1과 비슷하게 Game 2 대한 알고리즘 별로의 성능 차이도 아주 작다고 해석이 가능하다.

Game 3

다음은 Game 3의 3개의 알고리즘에 대한 관찰기간 및 이탈예측

기간 별 이탈 예측 모델의 AUC이다 [그림 95, 그림 96, 그림 97].

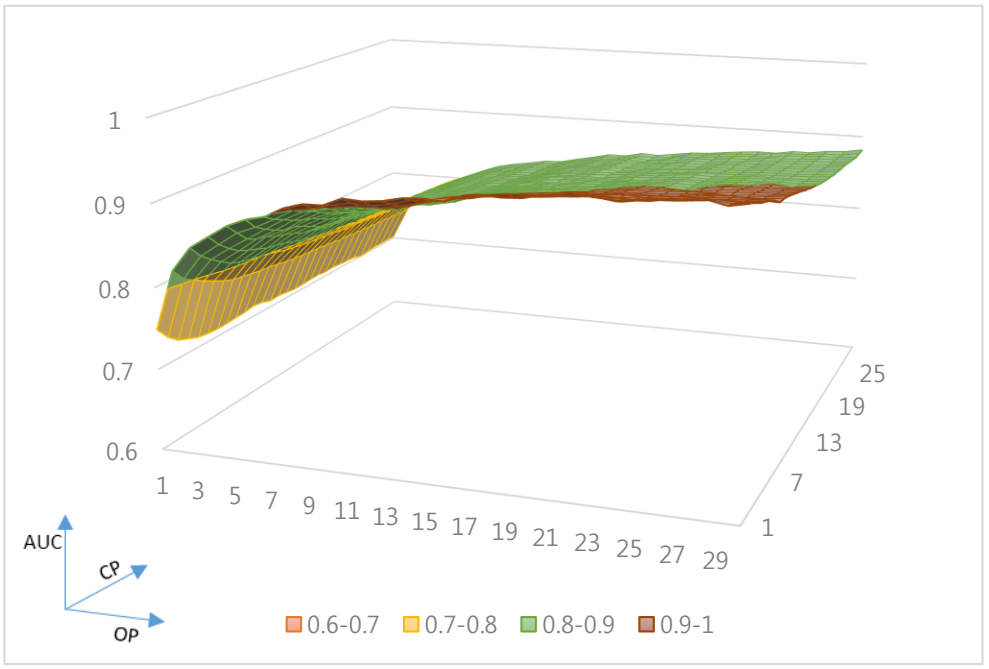

그림 95 Game 3 Gradient boosting 결과

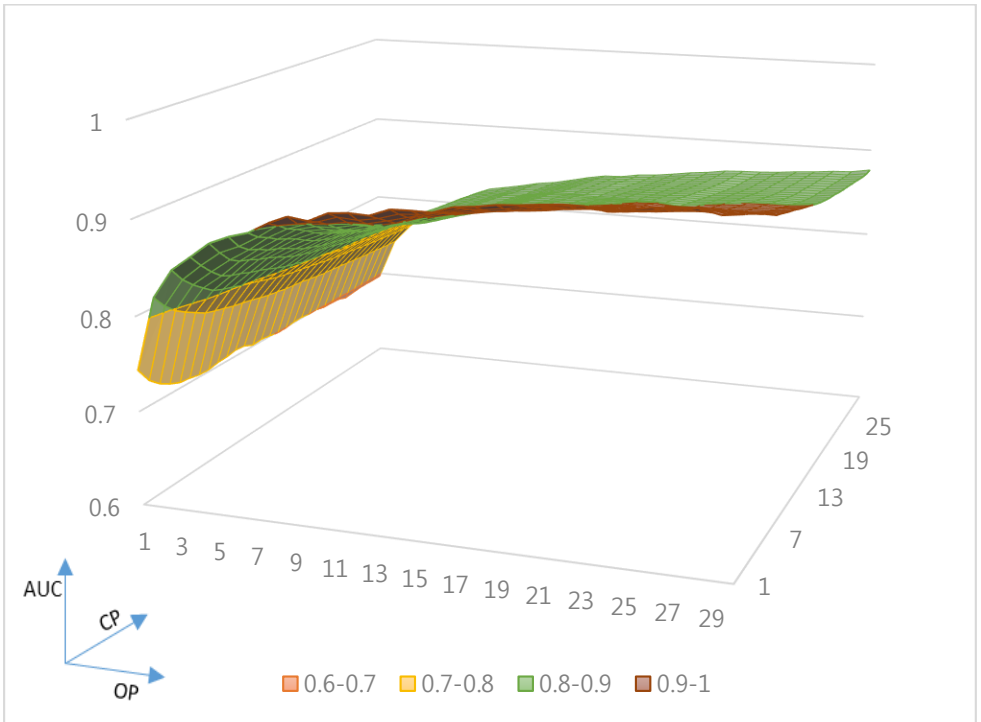

그림 96 Game 3 Logistic regression 결과

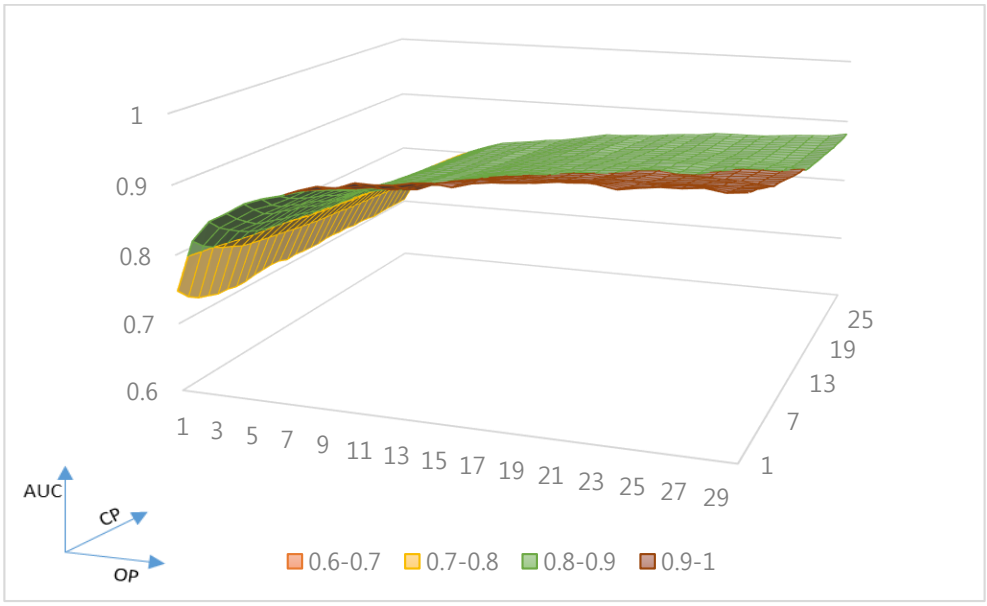

그림 97 Game 3 Random forest 결과

아래 세 그래프들은 Game 3의 각 알고리즘 별 구체적인 차이를 나타는 그래프들이다. [그림 98]의 AUC는 Gradient boosting 모델의 AUC에 Logistic regression 모델의 AUC을 뺀 수치, [그림 99]의 AUC는 Gradient boosting 모델의 AUC에 Random forest 모델의 AUC을 뺀 수치, [그림 100]은 Logistic regression 모델에 Random forest모델을 뺀 수치이다.

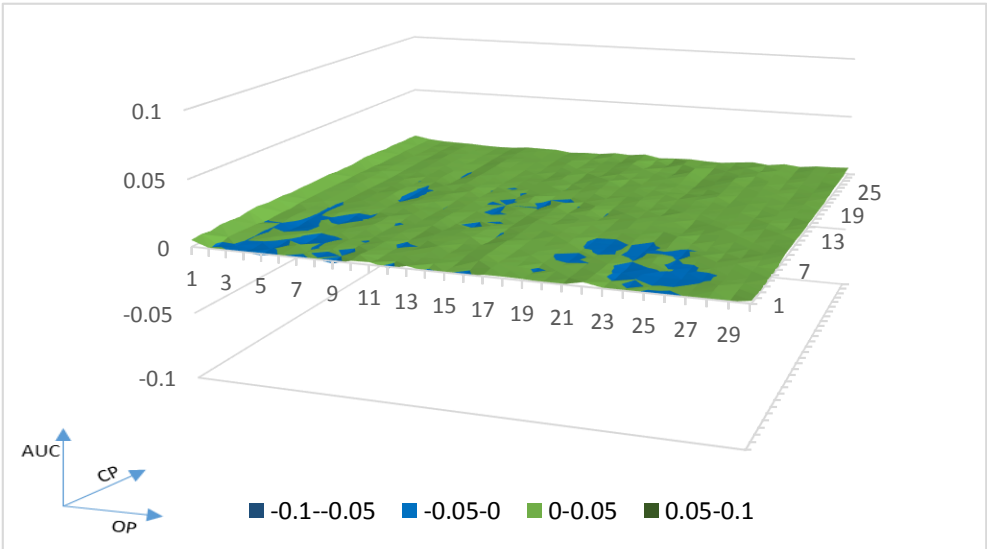

그림 98 Game 3 Gradient boosting과 Logistic regression의 AUC

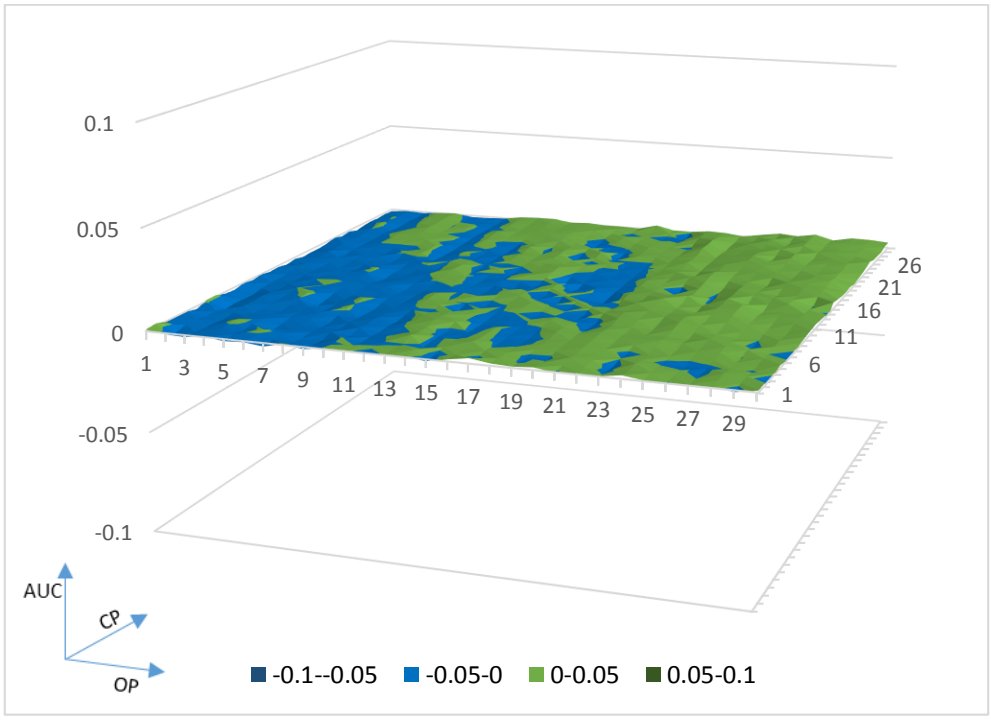

그림 99 Game 3 Gradient boosting과 Random forest의 AUC 차이

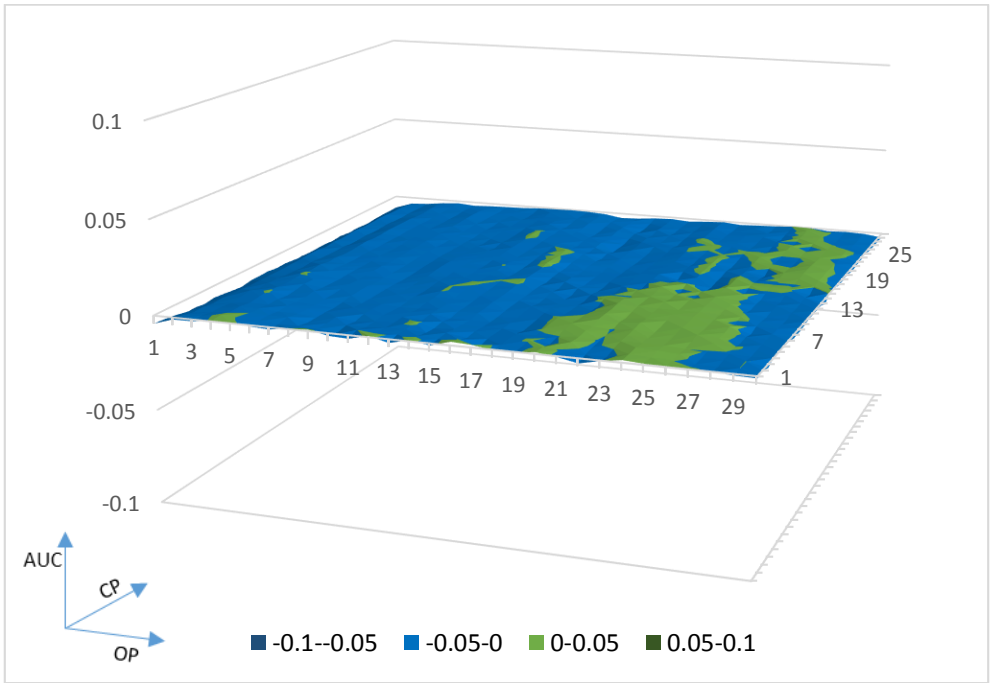

그림 100 Game 3 Logistic regression과 Random forest의 AUC 차이

위에서 Game3의 세개의 알고리즘 중 두개씩 짝을 지어 AUC값

차이를 비교 해보았다. 세개 알고리즘에 대한 차이를 한번에 보기 위하여, 세개의 알고리즘에 대한 표준 편차 그래프를 다음과 같이 그려보았다[그림 101].

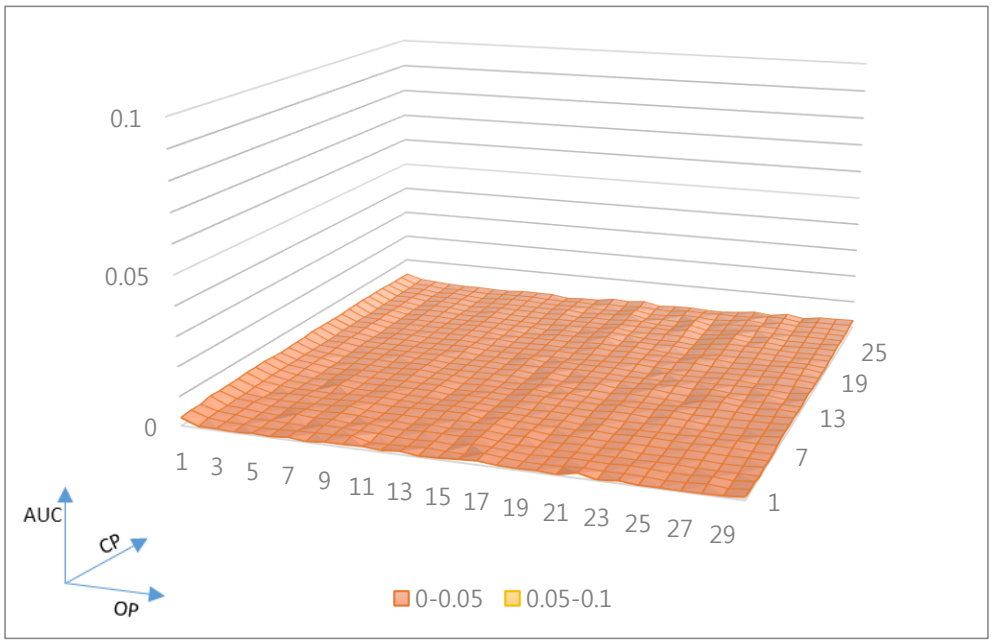

그림 101 Game 3 세 알고리즘의 표준편차

세개의 알고리즘에 대한 그래프의 형태도 비슷하게 나왔다. 또한, 알고리즘 중 관찰 기간이 9일 이하인 AUC값들은 Random forest의 성능이 가장 높았으며 9일 이후의 AUC값들은 Gradient boost의 성능이 가장 높았다. 그러나 전체 게임의 표준 편차의 크기를 살펴보면 대부분 0.01 미만으로 분포되어있다. 이는 실제로 Game 1, Game 2와 비슷하게 Game 3 대한 알고리즘 별로의 성능 차이도 아주 작다고 해석이 가능하다.

제 6 장 결 론

제 1 절 연구 결과의 요약

본 연구에서는 총 3종의 게임 데이터를 가지고 유저 이탈 예측 모델을 설계하고, 예측 모델을 도출하였으며, 예측 결과에 대한 성능평가를 하였다. 이러한 일련의 데이터 분석 과정동안 이탈 예측 모델에서 사용할 Feature들에 대한 개별 분석과 예측 성능에 끼치는 영향에 대해서 알아보았다. 또한, 관찰 기간과 이탈예측 기간의 변화에 따라서 예측 성능에 어떤영향을 끼치는지 분석하였다. 그리고 세개의 알고리즘들(Gradient boost, Logistic regression, Random forest)에 대한 유저 이탈 예측 모델의 예측성능을 비교해보았다. 다음은 본 연구에서 정한 세가지 연구문제와 관련된 결론이다.

연구문제 1. 유저 이탈 예측 모델에 있어서 Feature들이 예측 성능 향상에 얼마나 기여하는가?

본 연구에서는 유저 이탈 예측 모델에 있어서 Feature들이 예측 성능 향상에 얼마나 기여하는지에 대해서 살펴보았다. 총 14가지의 Feature들을 정의 하였고 개별적으로 기본적인 통계를 통해 알아보았다. 또한, 최종적으로 총 6가지의 방법으로 Feature들의 순위를 매겼고, 6개 순위의 평균을 통해서 게임 별 Feature들의 최종 순위를 결정하였다.

세개의 게임 공통으로 activeDuration, playCount가 1위, 2위를 차지하였다. 이를 가지고 게임 플레이 기간이 짧고, 게임 플레이 횟수가 적을수록 이탈을 하게된다는 일반적인 해석이 가능하다. 또한, 유저 이탈 예측을 할 시 게임플레이 기간과 게임플레이 횟수만으로도 일반적인 성능의 예측 모델을 만들 수 있다는 해석이 가능하다.

특정 게임에만 높은 순위인 Feature 들도 있다. bestScore (Game 1), consecutivePlayRatio (Game 1, Game 2), worstScore (Game 3), bestScoreIndex (Game 3), purchaseCount(Game 2), bestPurchase(Game 2)들이 높은 순위의 Feature들이다. 또한, 순위가

높은 Feature들에 대해서 이탈여부와의 correlation을 가지고 해석이 가능하다. 최고 점수가 낮고, 구매한 횟수가 적고, 비싼 상품을 구매한적이 없고, 최고 점수가 플레이 후반에 있는 유저일수록 이탈을 한다는 해석이 가능하다. 또한, 위 Feature들을 사용한다면 앞서 말했던 activeDuration과 playCount가 가지고 있지 않은 정보들을 가지고 있기 때문에 예측 성능의 향상을 가져다 준다는 해석이 가능하다.

또한, meanScore는 세개의 게임 모두 최하위를 기록 하였다. correlation도 세게임 모두 제일 낮게 나왔다. 직관적으로는 평균점수가 낮은 유저들은 점수가 부진하기 때문에 게임이 재미가 없어 이탈을 한다라는 일반적인 예측이 가능하지만, 세개의 게임 모두 유저의 평균점수가 높고 낮음이 이탈과는 상관이 없고, 심지어 이탈 예측에도 도움이 되지 않는다는 결과가 나왔다. 다시 해석하자면 유저들마다 게임의 적응력과 기본 실력, 게임 매커니즘에 대한 이해도 등이 다른데, 단순히 평균 게임 점수만을 가지고 이탈 예측을 할 수는 없다라는 해석이 가능하다. 이렇게 meanScore의 Feature 의 속성을 알아냄으로써 유저 이탈 예측 하는데 있어서 또 다른 수고를 덜어줄 수 있다.

Feature를 생성하는 데이터의 관점에서도 살펴볼 수 있다. Feature들을 만든 데이터는 플레이 횟수, 점수, 플레이 기간, 상점 구매(Game 2), 승률(Game 3) 데이터이다. Feature들의 순위를 보면, 플레이 기간과 플레이 횟수로 제작한 Feature들이 가장 좋은 성능을 보였고 위 Feature들은 상점 구매 데이터나 승률 데이터로 만든 Feature들보다 게임에 대한 범용성이 좀더 높은 Feature들이다. 따라서 게임 데이터에 대한 분석 및 예측 시, 플레이 횟수, 플레이 기간 관련된 Feature들에 대한 분석을 우선적으로 관찰한다면 좀더 효율적인 데이터 분석을 할 수 있다는 해석이 가능하다.

연구문제 2. 유저 이탈 예측에 있어서 관찰 기간 및 이탈예측 기간은 예측 성능에 어떠한 영향을 미치는가?

유저 이탈 예측 모델을 설계하는데 있어서 유저 이탈을 정의하는 것은 매우 중요한일이다. 기존 연구에서는 유저 이탈에 대해

제한된 정의를 가지고 연구가 진행되었다는 한계점이 있다. 이에 본 연구에서는 유저 이탈에 대한 정의를 관찰 기간과 이탈 예측 기간이라는 개념을 통해서 정확하게 정의 하였으며, 관찰 기간과 이탈예측 기간에 따른 예측 성능에 대해서도 연구를 하였다.

본 연구에서 관찰 기간은 1일부터 30일 까지로 잡고, 이탈예측 기간도 1일부터 30일로 잡았다. 그리고 관찰 기간과 이탈예측 기간을 변화해가면서 게임 별 900개의 데이터 테이블을 생성을 하였고, 총 2,700개의 데이터 테이블을 가지고 유저 이탈 예측 모델을 설계 및 생성을 하였고 예측 성능 평가까지 하였다. 예측 모델 별로 10-fold cross validation 검증기법을 통해 ROC curve의 AUC를 구하였고 다양한 그래프를 통해서 결과를 분석하였다.

세개의 게임 모두 관찰 기간이 길어지면 예측 성능이 좋아졌고, 이탈 예측 기간이 길어질수록 예측 성능이 나빠졌다. 즉, 이탈 예측 성능은 유저의 플레이에 대한 관찰기간을 많이 둘수록 많은 정보가 쌓이게 되고 이를 통해서 좋은 예측이 가능하고, 예측해야 하는 기간이 길어지면 점점 예측에 대한 불확실성이 커지게 되고, 예측 성능도 동시에 나빠지게 된다는 해석이 가능하다.

또한, 이탈 예측 하는데 있어서 관찰 기간과 이탈예측 기간에 대한 적당한 기간의 제안이 가능하다. 세개의 게임 모두 예측을 할 때 관찰 기간과 이탈예측 기간이 모두 초반 기간 이전일 때 관찰 기간이나 이탈예측 기간이 변화함에 따라서 예측 성능에 대한 변화가 컸고, 초반 기간 이후 기간에는 이전 기간에 비해 예측 성능에 대한 변화가 적었다. 이는 앞으로 이탈 예측을 할 때, 관찰 기간과 이탈예측 기간에대한 종속성없이 안정적인 예측을 하기 위해서는 적어도 두 기간을 게임의 중반 기간 이상으로 잡고 예측 모델을 설계하면 안정적인 예측을 할 수 있다는 해석이 가능하다. 또한 게임이 처음 출시되어 다이내믹한 일들이 벌어지고 이탈예측의 성능이 더욱 중요할 수 있는 시기에 상대적으로 성능이 떨어진다는 해석도 할 수 있다.

연구문제 3. 유저 이탈 예측 모델에 있어서 알고리즘별 예측 성능은 어떠한가?

본 연구에서는 세개의 알고리즘(Gradient boosting, Logistic regression, Random forest)을 가지고 유저 이탈 예측 모델을 연구하였다. 관찰 기간과 이탈예측 기간을 변화해가면서 게임 별로 만든 2,700개의 데이터 테이블을 가지고, 각 알고리즘 별로 이탈예측 모델을 설계 및 생성하였고 ROC curve의 AUC를 통해서 알고리즘 별 비교를 하였다.

Game 1에서는 관찰 기간을 첫날부터 약 20일까지로 정의한 예측 모델에 대해서는 Gradient boosting의 성능이 가장 좋았으며, 약 20일 이후의 예측 모델의 성능은 Logistic regression의 성능이 가장 좋았다. Game 2에서는 관찰 기간을 첫날부터 약 16일까지로 정의하였을 때 Gradient boosting의 성능이 가장 좋았으며, 약 16일 이후로 정의하였을 때엔 Logistic regression의 성능이 제일 좋았다. Game 3에서는 관찰 기간을 첫날부터 약 9일까지로 정의했을 때 Random forest의 성능이 제일 좋았으며, 약 9일 이후엔 Gradient boosting의 성능이 제일 좋았다. 대체적으로 세개의 게임 모두 Gradient boosting의 성능이 좋게 나왔다.

그러나 게임 별 세 알고리즘의 예측 모델의 성능 결과에 대한 AUC의 표준편차를 관찰 기간 및 이탈예측 기간 별로 구해본 결과 대부분의 표준편차가 0.01미만을 기록하였다. 이는 3개 게임의 유저 이탈 예측 모델의 알고리즘 별 차이가 거의 없다는 해석이 가능하다.

제 2 절 연구의 의의

현재 게임 시장은 모바일 게임을 선두로 점점 커져가고 있다. 점점 더 많은 유저가 게임을 즐길 것이며, 당연히 처음으로 게임을 즐기는 유저들도 많을 것이다. 캐주얼 게임은 대부분의 연령대가 남녀노소 불문하고 짧은 시간에 즐길 수 있기에, 처음으로 게임을 접하는 사람들이 접근하기 아주 쉽고, 이에 앞으로도 꾸준히 캐주얼 게임이 만들어 질 것이다. 또한, 큰 기업들은 지속적으로 새롭게 출시하는 게임들 사이에서 살아남으려 지속적으로 여러 광고 채널에서 대규모의 광고비를 집행하면서 유저를 확보하려 하고 있고, 작은 기업들은 이런 대기업들 사이에서 살아남으려 또 다른 광고 채널에서 적은 광고비를 빠듯하게 집행하면서 유저들을 획득하려고 할 것이다. 이렇게 앞으로도 시장은 지속적으로 커질 것이고 이에 비례하여 경쟁도 더 심해질 것이다.

그러나 위와 같은 빠르고 규모가 큰 시장상황에 비해, 게임 관련 연구에 대한 학술적인 접근이 많이 부족한 상황이다. 특히 유저 확보를 위해 무한 경쟁을 하고 있는 기업 입장에서도 제대로 된 분석 없이 광고나 프로모션만으로 유저를 획득하려 하는 데는 한계가 있다.

이러한 게임 업계의 상황에서 본 연구가 가져다 줄 기여는 확실하다. 신규 유저 획득을 위해 무한 경쟁을 하고 있는 게임 업계 상황에 유저 retention(유지)의 면에서 새로운 패러다임을 제시할 뿐만 아니라, 기존에 유저 retention에 대해 불명확한 분석을 하고 있는 곳에서도 데이터 분석에 기반한 과학적 유저 이탈 예측 연구에 대한 일종의 튜토리얼을 제시함으로써, 무한 경쟁을 하고 있는 업계에 분석의 틀을 마련해 줄 수 있는데 본 연구의 의의가 있다.

제 3 절 연구의 한계 및 제언

본 연구에서는 총 3개의 게임에 대한 데이터를 사용하고 있다. 그 중 한 개는 연구자 본인이 직접 개발하고 5년동안 서비스를 한 게임이고 나머지 2개는 공개적으로 오픈된 데이터이다. 이러한 오픈 데이터는 온라인 상에서 거의 찾아보기가 힘들고, 획득에 대한 진입장벽이 매우 높은데, 이런 데이터를 얻기 위해서는 실제로 산업에서 실무를 하지 않으면 얻을 수 없기 쉽다. 왜냐하면 게임 데이터는 유저들의 개인 정보뿐만 아니라 기업 자체의 과거, 현재의 매출과 앞으로 미래의 매출까지 예측이 가능한 민감한 데이터이기 때문에다. 이런 민감한 데이터는 기업에서 공개하기 매우 꺼리고, 공개가 되더라고 이미 성공한 게임의 데이터나 인기가 많은 게임의 데이터가 아닌 실패한 게임이거나 인기가 없는 게임인 경우가 더 많다. 기업이 아니라 개인이나 소규모에서 만든 게임인 경우엔 더 심각하다. 현재 게임 시장 상황이 소규모 게임에게 관대한 편이 아니며 그렇기 때문에 게임 데이터를 쌓기는 힘들다. 만약 게임의 인기가 많아지더라도 소규모에서 만든 게임이기 때문에 게임 개발에만 급급하여 데이터 분석에 대한 설계 없이 개발의 필요에만 맞는 데이터 수집만 하고 있는 경우가 많은 상황이다.

이러한 한계를 극복하기 위해서는 게임 데이터 분석에 대한 업계의 이해를 높이고 데이터 분석을 통해 게임 업계에서 추가적인 수익을 올리는 사례를 지속적으로 만들어야 한다. 현재 데이터 분석으로 인해 기업에 추가적으로 수익을 일으키는 사례는 점점 많아지고 있는 실정이다. 그러나 아직까진 국내 게임 업계에서는 데이터 분석을 통한 그렇다할 성과를 내지 못하고 있다. 지속적으로 게임 데이터 분석에 대한 업계의 이해를 높이고, 큰 회사에서부터 데이터 분석을 통한 추가 수익 획득에 대한 많은 관심을 가져 준다면 그러인한 파급 효과는 클 것이다. 이로인해 게임 관련 데이터 분석을 위한 회사 내, 외부 지원을 한다면 회사 밖의 게임 분야 외의 데이터 분석 전문가들까지도 게임 업계 쪽으로 관심을 가지게 되고 동시에 많은 학계의 연구가 이뤄질 것이다. 당연히 게임 데이터들은 공개적으로 많이 풀리게 될 것이고 간단한 데이터 셋으로 게임 데이터 분석하는 것에 대해 배우는 사람들도

많아지게 될 것이다.

위와같이 데이터 분석에 관한 관련 게임 업계의 이해를 높이고 실제 사례들을 많이 만들기 위해서는 본 연구의 주제인 유저 이탈 예측 연구가 추가적으로 더 이뤄져야한다. 유저 retention 비용이 새로운 유저 획득비용의 20%이고, 유저 이탈 예측 연구를 통해 제대로 유저를 예측하고 retention을 올려 매출을 상승시키는 사례가 나온다면 기업에서도 관심을 가질 수 있다[16]. 따라서 후속연구에서는 다양한 장르에 대한 적용을 위한 더 많은 게임 데이터를 가지고, 정형화된 유저 이탈 예측 연구 모델을 설계하고, 예측한 결과를 실제 산업에 적용 하여, 실제 수익화에 도움되기를 기대해본다. 그럼으로써 앞으로 게임 유저 이탈 연구 환경뿐만 아니라 게임 데이터 분석 환경에 대한 더 나은 상황을 제시해 줄 수 있을 것이다.

참고문헌

[1] 김주영, 조찬영, 장세정 & 윤은정. (2015). 2015 인터넷이용실태조사. 한국인터넷진흥원.

[2] 임재명, 장세정, 김민영 & 이정환. (2014). 2014 인터넷이용실태조사. 한국인터넷진흥원.

[3] 임재명, 유지열, 장세정, 이정환 & 유재민. (2013). 2013 인터넷이용실태조사. 한국인터넷진흥원.

[4] 김영수, 윤치호, & 김찬대. (2011). 2011 한일 게임이용자 조사보고서. 한국콘텐츠진흥원 연구보고서 11-10.

[5] 조영기, 이은숙 & 이경영. (2015). 게임이용자 실태조사보고서. 한국콘텐츠진흥원 연구보고서 15-11

[6] 이기현, 윤호진, 조영기, & 최원진. (2014). 게임이용자 조사보고서. 한국콘텐츠진흥원 연구보고서 14-13.

[7] 나스미디어. (2015). 한국 모바일 게임 시장 분석. http://www.slideshare.net/nasmedia/mobile-issue-report.

[8] Runge, J., Gao, P., Garcin, F., & Faltings, B. (2014, August). Churn prediction for high-value players in casual social games. In Computational Intelligence and Games (CIG), 2014 IEEE Conference on (pp. 1-8). IEEE.

[9] The shape of code. Aggregate player preference for the first 20 building created in Illyriad. (2015, June 7), Retrieved July 18, 2016, from http://shape-of-code.coding-guidelines.com/2015/06/07/aggregate-player-preference-for-the-first-20-building-created-in-illyriad/.

[10] Tagpro analytics. Science or how to analyse raw data yourself. (n.d.). Reteived July 18, 2016, from https://tagpro.eu/?science/.

[11] FreeToPlaybiz. The Business and Design of Free-To-Play Games. (2007, August 8). Retrieved March 15, 2015, from http://freetoplay.biz/2007/08/02/.

[12] 김인재, “통계로 보는 콘텐츠 산업-주요 국가 모바일게임 이용자 특성 비교”, 15-03, 2015

[13] Fowlkes, A. J., Madan, A., Andrew, J., & Jensen, C. (1999). The effect of churn on value: An industry advisory.

[14] Hadiji, F., Sifa, R., Drachen, A., Thurau, C., Kersting, K., & Bauckhage, C. (2014, August). Predicting player churn in the wild. In Computational Intelligence and Games (CIG), 2014 IEEE Conference on (pp. 1-8). IEEE.

[15] Borbora, Z., Srivastava, J., Hsu, K. W., & Williams, D. (2011, October). Churn prediction in mmorpgs using player motivation theories and an ensemble approach. In Privacy, Security, Risk and Trust (PASSAT) and 2011 IEEE Third Inernational Conference on Social Computing (SocialCom), 2011 IEEE Third International Conference on (pp. 157-164). IEEE.

[16] L. Luna. (1998). Churn is epidemic. Radio Commun.

[17] 천연비, 장성균, & 우탁. (2012). 게임 메커니즘에 따른 스마트폰 게임 분류 연구. Journal of Korea Game Society, 12(6), 15-24.

[18] Seung-Wook Kim, Woochang Hyun, Sung-Heum Lim, & Tack woo. (2013). Log Data Analysis for Comparing Conventional Game Controller and Motion Based Game Controller. International Conference on Multimedia Information Technology and Application.

[19] Feng, W. C., Brandt, D., & Saha, D. (2007, September). A long-term study of a popular MMORPG. In Proceedings of the 6th ACM SIGCOMM Workshop on Network and System Support for Games (pp. 19-24). ACM.

[20] Bauckhage, C., Kersting, K., Sifa, R., Thurau, C., Drachen, A., & Canossa, A. (2012, September). How players lose interest in playing a game: An empirical study based on distributions of total playing times. In Computational Intelligence and Games (CIG), 2012 IEEE Conference on (pp. 139-146). IEEE.

[21] Williams, D., Yee, N., & Caplan, S. E. (2008). Who plays, how much, and why? Debunking the stereotypical gamer profile.

Journal of Computer-Mediated Communication, 13(4), 993-1018.
[22] Keaveney, S. M., & Parthasarathy, M. (2001). Customer switching behavior in online services: An exploratory study of the role of selected attitudinal, behavioral, and demographic factors. Journal of the Academy of Marketing Science, 29(4), 374-390.
[23] Coussement, K., & Van den Poel, D. (2008). Churn prediction in subscription services: An application of support vector machines while comparing two parameter-selection techniques. Expert systems with applications, 34(1), 313-327.
[24] Xie, Y., Li, X., Ngai, E. W. T., & Ying, W. (2009). Customer churn prediction using improved balanced random forests. Expert Systems with Applications, 36(3), 5445-5449.
[25] Anil Kumar, D., & Ravi, V. (2008). Predicting credit card customer churn in banks using data mining. International Journal of Data Analysis Techniques and Strategies, 1(1), 4-28.
[26] Morik, K., & Köpcke, H. (2004). Analysing customer churn in insurance data-a case study. In Knowledge Discovery in Databases: PKDD 2004 (pp. 325-336). Springer Berlin Heidelberg.
[27] Runge, J., Gao, P., Garcin, F., & Faltings, B. (2014, August). Churn prediction for high-value players in casual social games. In Computational Intelligence and Games (CIG), 2014 IEEE Conference on (pp. 1-8). IEEE.
[28] Kawale, J., Pal, A., & Srivastava, J. (2009, August). Churn prediction in MMORPGs: A social influence based approach. In Computational Science and Engineering, 2009. CSE'09. International Conference on (Vol. 4, pp. 423-428). IEEE.
[29] Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. Journal of machine learning research, 3(Mar), 1157-1182.
[30] Unix_time. (n.d.). In Wikipedia. Retrieved October 14, 2009, from http://en.wikipedia.org/wiki/Unix_time

[31] Blum, A. L., & Langley, P. (1997). Selection of relevant features and examples in machine learning. Artificial intelligence, 97(1), 245-271.

[32] Scoring System. (n.d.). In Tagpro wiki. Retrieved March, 2016, from https://www.reddit.com/r/TagPro/wiki/score

[33] Gradient_boosting. (n.d.). In Wikipedia. Retrieved May 8, 2016, from http://en.wikipedia.org/wiki/Gradient_boosting

[34] Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. Annals of statistics, 1189-1232.

[35] Lawrence, Rick, et al. "Classification of remotely sensed imagery using stochastic gradient boosting as a refinement of classification tree analysis." Remote sensing of environment 90.3 (2004): 331-336.

[36] Friedman, J. H. (2002). Stochastic gradient boosting. Computational Statistics & Data Analysis, 38(4), 367-378.

[37] Li, C. (n.d.). A Gentle Introduction to Gradient Boosting. Retrieved March, 2016, from http://www.chengli.io/tutorials/gradient_boosting.pdf.

[38] Vapnik, V. (2000). The Nature of Statistical Learning Theory. Information Science and Statistics.Springer-Verlag.

[39] Hosmer Jr, D. W., & Lemeshow, S. (2004). Applied logistic regression. John Wiley & Sons.

[40] James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An introduction to statistical learning (p. 6). New York: Springer.

[41] Liaw, A., & Wiener, M. (2002). Classification and regression by Random forest. R news, 2(3), 18-22.

[42] Random_forest. (n.d.). In Wikipedia. Retrieved May 8, 2016, from http://en.wikipedia.org/wiki/Random_forest

[43] Breiman, L. (2001). Random forests. Machine learning, 45(1), 5-32.

[44] Hastie, Trevor; Tibshirani, Robert; Friedman, Jerome (2008). The Elements of Statistical Learning (2nd ed.). Springer. 587-

588.

[45] Cross-validation_(statistics). (n.d.). In Wikipedia. Retrieved October 14, 2009, from http://en.wikipedia.org/wiki/ Cross-validation_(statistics)

[46] McLachlan, G., Do, K. A., & Ambroise, C. (2005). Analyzing microarray gene expression data (Vol. 422). John Wiley & Sons.

[47] Hanley, J. A., & McNeil, B. J. (1982). The meaning and use of the area under a receiver operating characteristic (ROC) curve. Radiology, 143(1), 29-36.

[48] Fawcett, T. (2006). An introduction to ROC analysis. Pattern recognition letters, 27(8), 861-874.

[49] Bamber, D. (1975). The area above the ordinal dominance graph and the area below the receiver operating characteristic graph. Journal of mathematical psychology, 12(4), 387-415.

[50] Pencina, M. J., D'Agostino, R. B., & Vasan, R. S. (2008). Evaluating the added predictive ability of a new marker: from area under the ROC curve to reclassification and beyond. Statistics in medicine, 27(2), 157-172.

[51] Precision_and_recall. (n.d.). In Wikipedia. Retrieved October 14, 2009, from http://en.wikipedia.org/wiki/Precision_and_recall

[52] Fischer, J. E., Bachmann, L. M., & Jaeschke, R. (2003). A readers' guide to the interpretation of diagnostic test properties: clinical example of sepsis. Intensive care medicine, 29(7), 1043-1051.

[53] Guyon, Isabelle, and André Elisseeff. "An introduction to variable and feature selection." Journal of machine learning research 3. Mar (2003): 1157-1182.

Abstract

Churn Prediction of Casual Game

Seungwook Kim
Program in Digital Contents and Information Studies
Department of Transdisciplinary Studies
The Graduate School
Seoul National University

Recently, as the scale of the game market has increased, the casual game, which now takes the largest share of the game market, has taken the center stage. Accordingly, the game industry is releasing many casual games these days with a heavy focus on attracting more users to maximize its profit. Lately, however, the industry has realized that the number of additional users it can attract is becoming limited; thus, the importance of existing user retention has been emphasized over new user acquisition.

In this research, a churn prediction model using user data is suggested, solving a limitation in existing churn research and providing insights by researching a series of churn prediction processes. The research uses data on three casual games, including a game designed for the present research, to examine the above circumstance. Through the data, which traces user acquisition and churn, this research contributes to the field of user retention in the game industry by presenting churn prediction models and their analyses. This research is of importance because it can provide

data-driven insight and improve game industry profit.

The research proposes churn prediction model using user data, analyzes the characteristics of the features used in the prediction model and examines prediction model performance. It also examines the effect of changing the observation period and churn prediction period on prediction model performance. Lastly, it compares the performance of the proposed prediction model between three algorithms.

This research offers a definition of churn, collects the raw data to use in the research through data-preprocessing steps, and processes the data to make it easily applicable to the algorithms. After that, gradient boosting, logistic regression, and random forest prediction modeling are designed for each of the three games by using 10 common features and 4 exclusive features, thatcan be applied to only a specific game. Finally, the performance of the churn prediction model is estimated with an ROC (Receiver Operating Characteristic) AUC (Area Under the Curve) through 10-fold cross-validation.

The results of this research show that the activeDuration and playCount features have the largest effect on the performance of the prediction model, and the bestScore, consecutivePlayRatio, worstScore, bestScoreIndex, purchaseCount and bestPurchase features have an additional effect on the performance. Moreover, a longer observation period and a shorter churn prediction period lead to better performance of the prediction model. Furthermore, when the observation period and the churn prediction period are in the early phase of the game's life cycle, the variance of the prediction performance is high than the later phase. Therefore, it can produce

reliable results when the observation period and churn prediction period are defined after the early phase. Finally, the gradient boosting algorithm has the best prediction performance among the three algorithms, but the difference in prediction performance is marginal.

Keywords: Casual Game, Churn prediction, User analysis, Gradient boosting, Logistic regression, Random forest
Student Number: 2012-23860